VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2018 Matej Boszorád

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

SLEDOVÁNÍ OBJEKTŮ VE VIDEU
OBJECT TRACKING IN VIDEO

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Matej Boszorád**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Martin Rajnoha**
SUPERVISOR

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Matej Boszorád ***ID:*** 186035
Ročník: 3 ***Akademický rok:*** 2017/18

NÁZEV TÉMATU:

Sledování objektů ve videu

POKYNY PRO VYPRACOVÁNÍ:

Prostudujte a popište metody pro sledování objektů ve videu. Zaměřte se na neučící se algoritmy řešící zejména přiřazovací problém. Vytvořte algoritmus pro sledování objektů ve videu. Algoritmus musí být schopen pracovat s více než jedním parametrem objektu. Výsledky vhodně reprezentujte a demonstrujte na příkladech.

DOPORUČENÁ LITERATURA:

[1] KUHN, Harold W. The Hungarian method for the assignment problem. Naval Research Logistics (NRL), 1955, 2.1-2: 83-97.

[2] YILMAZ, Alper; JAVED, Omar; SHAH, Mubarak. Object tracking: A survey. Acm computing surveys (CSUR), 2006, 38.4: 13.

Termín zadání: 5.2.2018 ***Termín odevzdání:*** 29.5.2018

Vedoucí práce: Ing. Martin Rajnoha
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Táto bakalárska práca sa zaoberá problematikou sledovania viacerých objektov vo videu, so zameraním na neučiace algoritmy. Prvá kapitola predstavuje teoretickú časť práce, v ktorej sú popísané jednotlivé sledovacie metódy. Tieto algoritmy sú rozdelené podľa vlastností, ktoré používajú pre správne sledovanie. V tejto sekcií je popísaný princíp sledovacích metód mean-shift, škálovo invariantnej tranformácie objektu, Kalmanovho filtru, časticového filtru a Gáborovej vlnkovej transformácie. V kapitole je taktiež riešený problém priradenia, ktorý sa zaoberá hlavne Maďarskou metódou. Ďalšia časť kapitoly opisuje možnosti zlúčenia viacerých sledovacích metód, ktoré sú rozdelené podľa typu konštrukcie na paralelné, kaskádovité, vážené a diskriminatívne ohodnotenie s príkladmi. Rovnako je v tejto časti popísaná adaptívnosť sledovacieho systému. V práci sú ďaľej popísané problémy nastávajúce pri sledovaní a riešenie týchto problémov. Daná sekcia pozostáva z riešenia šumu obrazu, zmeny osvetlenia, zjavu a zániku objektu pričom sa práca sústreďuje predovšetkým na riešenie problému oklúzie dvoch objektov. V rámci praktickej časti je vytvorený algoritmus zložený z rozličných typov sledovania, ktorého výsledky sú následne porovnané s vybranými sledovacími systémami z benchmarku pre sledovanie viacerých objektov vo videu. Praktická časť zahŕňa použité nástroje a vysvetlenie návrhu sledovacieho systému, v ktorom sú popísané hlavné triedy a metódy použité pre tvorbu. Okrem toho je v tejto sekcí popísané paralelné zlučovanie a použitá adaptívnosť sledovania. Vo výsledkoch práce sa nachádza porovnanie použitia sledovacích techník oddelene a spolu. Pre porovnanie výsledkov boli použité videosekvencie so sledovaním chodcov a sledovanie tvárí. Vychádzali sme z predpokladu, že zlúčenie viacerých sledovacích systémov napomôže k zlepšeniu sledovania čo bolo potvrdené vo výsledkoch práce.

KĽÚČOVÉ SLOVÁ

Kalmanov filter, oklúzia, priraďovací problém, sledovanie objektov, škálovo invariantná transformácia vlastnosti, zmena k stredu

ABSTRACT

This bachelor thesis deals with the issue of tracking multiple objects in a video, specifically focusing on non-learning algorithms. The first chapter represents the theoretical part of the thesis, in which some of the often used tracking methods are described, such as mean-shift, scale-invariant object transformation, Kalman filter, particle filter and Gabor wavelet transformation. These algorithms are broken down by properties they use for proper tracking. The chapter also contains section assignment problem, which is mainly concerned with Hungarian algorithm. The next part describes options of merging multiple tracking methods that are broken down by construction type into parallel, cascade, weighted and discriminatory with example for each one. Moreover there is described adaptability of the tracking system. Bellow are described problems which may occur during tracking and possible solutions to them. This section consists of a solution of image noise, changes in illumination, appearance and extinction of an object, focusing mainly on solving the problem of object occlusion. Within the practical part is created algorithm composed of different types of tracking, the results of which are then compared with selected tracking systems from the multiple object tracking benchmark. The practical part includes the tools used and the explanation of the design, in which the main classes and methods used for the tracking are explained. Besides that, this section describes parallel merging and tracking adaptability . The results of the thesis contain a comparison of the use of tracking techniques separately and together. To compare the results, videos for pedestrian tracking and face tracking were used. This thesis was based on the assumption that merging multiple monitoring systems will help with the improvement of the tracking, which was confirmed by the results.

KEYWORDS

Assignment problem, Kalman filter, mean-shift, object tracking, occlusion, Sift

BOSZORÁD, Matej. *Sledovanie objektov vo videu*. Brno, Rok, 56 s. Bakalárska práca. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedúci práce: Ing. Martin Rajnoha

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Sledovanie objektov vo videu“ vypracoval(a) samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som sa poďakoval vedúcemu semestrálnej práce panu Ing. Martinovi Rajnohovi, za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno ....

podpis autora(-ky)

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ÚVOD

Sledovanie objektov v reálnom čase je kľúčovou úlohou pre najrôznejšie aplikácie, ako napríklad pre dohľadové systémy, interakciu s počítačom, rozšírenie reality alebo asistenciu riadenia. Táto práca je prevažne zameraná na sledovanie viacerých objektov, ktoré sú pri sledovaní reprezentované rôznymi vlastnosťami. Sledovanie objektov zahŕňa určenie všetkých výskytov objektov v slede snímkov. Pre správne sledovanie je potrebná identifikácia získaných detekcií k objektom pomocou vybraných vlastností. Práca je orientovaná na časť správneho priradenia detekcií k objektom. Cieľom tejto práce je vytvorenie algoritmu pre sledovanie objektov pomocou viacerých parametrov objektu.

Práca je rozdelená na časti zaoberajúce sa metódami sledovania, priraďovacím problémom a riešením problémov týkajúcich sa sledovania. Skladá sa zo všeobecného úvodu do sledovania a následného opisu jednotlivých krokov pri tvorbe algoritmu pre sledovanie objektov. Prvá kapitola je venovaná preberaným metódam sledovania využívajúcim vlastnosti objektov, ako napríklad farby objektu, body záujmu alebo ich postup v obraze. Nasledujúca časť rozoberá priraďovací problém a použité riešenia pre sledovanie pomocou viacerých parametrov. Tretia kapitola sa zameriava na problémy nastávajúce pri použití jednotlivých sledovacích metód a ich riešenie. Posledná časť je venovaná návrhu samotného riešenia, použitiu metód sledovania a zlúčeniu týchto metód. Tu sú vysvetlené triedy a metódy podstatné pre tvorbu sledovania. V záverečnej časti je zhrnuté vyhodnotenie programového riešenia na testovacích videách a ukážka riešenia prechodu dvoch objektov na tomto videu.

1 TEORETICKÁ ČASŤ ŠTUDENTSKEJ PRÁCE

Sledovanie objektu je jedným z najdôležitejších problémov v odbore počítačového videnia, pričom jeho aplikácie majú veľký dosah, napríklad pri dohľadových systémoch, interakcii s ľudmi alebo robotike. Hlavným cieľom sledovania je proces neustáleho určovania stavu objektu založenom na meraniach rôznych senzorov alebo odhadom pomocou rôznych metód [1]. V prípade, že sa objekt dostane von z dosahu senzorov, táto entita zanikne a v prípade, že sa dostane objekt do oblasti detekovanej senzormi, nová entita vznikne. V našom prípade sa senzorom myslí videokamera, ktorá však nedokáže získavať potrebné údaje alebo vlastnosti objektov pre spracovanie počítačovou technikou. Preto je nutné zavádzať rôzne metódy zisťovania stavu objektu, ktoré sú kľúčovými vo veľkom počte technických aplikácii. Hlavný stav riešený metódami sledovania je poloha v obraze. Nie vždy však tieto detekcie udajú konkrétne priradenie k objektu, a preto tu vzniká problém priraďovania, ktorý je v tejto práci riešený. Na začiatok je však nutné zistiť, ako tieto detekcie fungujú.

1.1 Metódy sledovania objektu

Jedným z hlavných problémov sledovania je reprezentácia objektu tak, aby bol čo najefektívnejšie rozlíšiteľný medzi dvoma sledovanými objektami. Táto sekcia rozoberá analýzu obrazu pomocou detektorov najlepších vlastností a deskriptorov[1], pre popis týchto detekcií . Cieľom tejto časti je lepšie pochopenie opisu objektu a jeho podiel práce pri sledovaní objektu.

1.1.1 Sledovanie pomocou farby

Hlavnou myšlienkou sledovania pomocou farby je porovnávanie histogramov dvoch rôznych detekcií obrazu. Histogram, graf vyjadrujúci distribúciu farby alebo jasu na x-ovej osi a na y-ovej osi, určuje počet výskytov danej farby alebo veľkosti jasu. Súčet distribúcii histogramu je

$$S = \sum_{u=1}^{m} q_u = 1, \tag{1.1}$$

kde q_u sú veľkosti jednotlivých distribúcii histogramu. Väčšinou sa histogram používa na úpravu prepalov na fotografiách, ale v našom prípade je histogram potrebný pre zistenie distribúcie farieb hľadaného objektu. Existujú rôzne typy vykresľovania obrazu ktoré sú popísané v [2], avšak popísané budú len hlavné:

[1] deskriptor-popis vlastnosti použiteľnej pre porovnanie

RGB,HSV,Luv,Lab,YCbCr

Modely obrazu uvedené v nadpise sa používajú na spracovanie farebného obrazu. Majú výhodu v mohutnosti informácie, ktorú prenášajú. Naopak použitie tohto typu modelov prináša nevýhodu v tom, že nemôžeme reprezentovať takýto model obrazu jednou maticou pre spracovanie filtrom, detekciou hrán a pod.

Odtiene sivej

Model odtieňov sivej, je tvorený primárne z jedného parametru, a tým je jas. Tento model je používaný hlavne pri filtrácii obrazu a vyhľadávaní hrán, nakoľko je možné reprezentovať obraz v odtieňoch sivej jednou maticou, pričom jas je jeho jediný parameter. Taktiež býva táto zložka obrazu najkvalitnejšia, pretože pri prenose farebné zložky prechádzajú zväčša stratovou kompresiou [3]. Nevýhodou tohto formátu obrazu je veľká senzitivita na zmenu jasu obrazu.

Nutnosťou pre sledovanie objektu pomocou farby je potreba zobraziť objekt ako určitú oblasť obrazu, napr. štvorec, aby sme mohli z danej oblasti vytvoriť histogram. Pre zvýšenie presnosti sledovania pomocou histogramu sa váhuje celý obraz Gaussovou maskou, ktorá je veľkosti stanovenej detekcie. Daný postup uprednostňuje časť obrazu, ktorá je najbližšie ku stredu obrazu. Ďalšou možnosťou, ako vylepšiť sledovanie, je odstránenie šumu v obraze, a to napr. Mediánovým filtrom. Taktiež je možné predom vyrezať časť obrazu, ktorá je siluetou objektu, a to pomocou detektora hrán, napr. Canny[4], bližšie popísaného v časti 1.4.1. V prípade, že je vytvorený histogram obrazu, je treba ho porovnávať medzi ostatnými oblasťami obrazu, a to pomocou Bhattacharyyovej vzdialenosti[5], ktorá sa počíta ako

$$f(y) = \sum_{u=1}^{m} \sqrt{p_u(y) q_u}, \tag{1.2}$$

kde $q_u(y)$ je u-ty prvok historgramu y-tej časti obrazu, vybraný pre porovnanie. q_u je u-ty prvok histogramu pre porovnávaného obrazu objektu. Po vytvorení koeficientov, ktoré sú mierou podobnosti, je vybraná časť obrazu, ktorá je najviac farebne podobná vzoru. Lepším použitím Bhattacharyyovej vzdialenosti je sledovanie pomocou zmeny k stredu(z angl. Mean Shift)[6].

Zmena k stredu

Ďalej nazývaný ako Mean shift je algoritmus sledovania pomocou vyhľadávania hustoty pravdepodobnosti podobnosti. Základnou myšlienkou metódy sledovania je vytvorenie bodov podobnosti v okolí objektu, ktorý je určovaný na základe Bhattacharyyovej vzdialenosti. Body podobnosti sú vytvorené z oblasti bodu podobnosti

nazývaného región záujmu, v ktorom je bod podobnosti umiestnený. Potom je vytvorený vektor, ktorý smeruje k zhluku týchto bodov vážených pomocou Bhattacharyovej vzdialenosti. Primárne je potrebné zadefinovať model objektu, ktorý je v Mean Shift reprezentovaný ako pravdepodobnosť q, farby $u = 1, ..., m$, ktorý sa počíta ako

$$q_u = C \sum_{i=1}^{n} k(||\mathbf{x}_i||^2) \delta(b(\mathbf{x}_i) - u), \tag{1.3}$$

kde C je normalizačná konštanta, k je profil kernelu, ktorý určuje príspevok na základe vzdialenosti od centra záujmu. δ je jednotkový impulz pričom

$$\delta(x) \begin{cases} 0 & \text{ak x=0} \\ 1 & \text{v ostatných prípadoch.} \end{cases} \tag{1.4}$$

Rovnako ako model objektu sú vytvorený aj kandidáti, ktorý sú porovnávaný podľa Bhattacharyyovho koeficientu

$$\rho(p(\mathbf{y}, q)) = \sum_{u=1}^{m} \sqrt{p_u(\mathbf{y}) q_u}, \tag{1.5}$$

kde $p_u(\mathbf{y})$ je u-ty prvok pravdepodobnosti kandidáta a q_u je u-ta pravdepodobnosť modelu objektu. Bhattacharyyov koeficient sa rozšíri na Taylorovu radu, je maximalizovaná pre vynímanie váh, ktoré sú priraďené pre jednotlivé oblasti podobností[7]. Tie sú vytvorené podľa

$$w_i(\mathbf{y}_0) = \sum_{u=1}^{m} \delta[S(\mathbf{x}_i) - u] \sqrt{\frac{q_u}{\hat{p}_u(\mathbf{y}_0)}} \tag{1.6}$$

pričom δ zistí, akou hodnotou prispieva daný bod x ku veľkosti u. $\hat{P}_u(\mathbf{y}_0)$ a q_u sú podiely v časti distribúcie u. Po vytvorení váh 1.6 je možné vytvoriť nové umiestnenie priradením vektoru k iniciálnej pozícii.

$$M_h(\mathbf{y}_0) = \left[\frac{\sum_{i=1}^{n_x} w_i(\mathbf{y}_0) \mathbf{x}_i}{\sum_{i=1}^{n_x} w_i(\mathbf{y}_0)} \right] - \mathbf{y}_0 \tag{1.7}$$

kde w_i sú jednotlivé váhy pre body, x_i sú body podobnosti a $\mathbf{y}_0$ je pozícia, z ktorej vychádzame [7].

$$\hat{\mathbf{y}} = \mathbf{y}_o + M_h(\mathbf{y}_0), \tag{1.8}$$

kde $\mathbf{y}_0$ je iniciálna pozícia a $M_h(\mathbf{y}_0)$ je vektor smerujúci k sledovanému objektu.

1.1.2 Zmena veľkosti detekcie

Zmena veľkosti detekcie je metóda sledovania založená na plynulej zmene objektu v obraze. Používa sa hlavne pri detekcií objektov rôznych veľkostí v značnej vzdialenosti od kamery. Táto metóda sledovania sa používa predovšetkým na určenie veľkosti kernelu objektu pri KLT algoritme[8], ktorý je z časti invariantný voči zmene

veľkosti, posunutia a iných jednoduchých zmien obrazu[9].V prípade veľkej zmeny veľkosti objektu môžu spôsobiť nesprávne priradenie k objektu. Problémom tohto detektoru môže byť frekvencia snímania kamery, nakoľko menej obrazov na porovnávanie znamená väčšie zmeny veľkosti obrazu. Táto metóda taktiež potrebuje reprezentovať objekt ako primitívny objekt, ktorého veľkosť sa dá jednoducho vypočítať.

1.1.3 Sledovanie pomocou zmeny pozície

Tento typ metód zahŕňa v sebe predpoklad plynulého postupu objektu v obraze. Táto metóda je jednoduchým riešením problému sledovania, avšak nezahŕňa v sebe riešenie prekrytia dvoch objektov. Najlepším riešením, čo sa týka tohto typu sledovania, je použitie predikcie a korekcie postupu objektu. Príkladom najčastejšie používanej sledovacej metódy pre riešenie prekrytia, resp. oklúzie je Kalmanov filter [10].

Kalmanov filter

Je odhad, ktorý predpokladá a koriguje stavy širokého rozsahu lineárnych procesov[11]. Pri sledovaní objektov Kalmanov filter koriguje stav pohybu, avšak je treba zaistiť dobrý model detekcie. Kalmanov filter priradí iniciálny stav objektu podľa miesta, v ktorom sa objekt nachádza, a jeho rýchlosť v určitom smere podľa miesta z predošlého obrazu. Potom je predikovaný nový stav

$$X_k = Ax_{k-1} + Bu_k + w_k = \begin{bmatrix} 1 & \Delta \mathrm{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ v_{0x} \end{bmatrix} + \begin{bmatrix} \frac{1}{2}\Delta t^2 \\ \Delta t \end{bmatrix} [a_{x_0}], \tag{1.9}$$

kde Ax_{k-1} je predikovaný stav na základe rýchlosti objektu a Bu_k je prídavok ku predikovanému stavu na základe zrýchlenia objektu. Následne je možné vytvoriť tabuľku kovariancie na základe procesných chýb, ktoré sú zapísané do matice kovariancie

$$P_{k-1} = \begin{bmatrix} \Delta x^2 & \Delta x \Delta v \\ \Delta x \Delta v & \Delta v^2 \end{bmatrix}, \tag{1.10}$$

pričom Δx a Δv sú rozdiely predikcie rýchlosti a pozície. Na základe tejto tabuľky potom je možné vytvoriť novú maticu, ktorá je predikciou z matice 1.10 kovariancie v ďalšom kroku

$$P_{k_p} = AP_{k-1}A^T + Q_k, \tag{1.11}$$

kde P_{k-1} je prvotná matica kovariancie a Q_k je chyba procesu kalkulácie predikovanej kovariancie matice, ktorú je možné nulovať. Po vytvorení predikovanej matice kovariancie je vytvorený Kalmanov zisk, ktorý je ukazovateľ chyby

$$K = \frac{P_{k_p}H}{HP_{k_p}H^T + R} \tag{1.12}$$

kde K je Kalmanov zisk, čo je matica, podľa ktorej je možné zistiť, aká veľká je chyba. Ďalej sa tu nachádza matica H vytvorená z hodnôt, ktoré predikujeme. R v rovnici pre výpočet Kalmanovho zisku je rozdiel detekovaných hodnôt od vypočítaných a zapísaných do matice

$$R = I \begin{bmatrix} \Delta x \\ \Delta v_x \end{bmatrix}. \tag{1.13}$$

Čím vačšia je matica K, tým je toto meranie presnejšie a naopak. Taktiež je možné danú maticu použiť pre výpočet novej pozície pre novú detekciu objektu

$$X_{k+1} = X_{k_p} + K(Y_k - HX_{k_p}) \tag{1.14}$$

kde X_{k_p} matica predicie objektu a Y_k je nové pozorovanie zapísané do matice. Po vytvorení nového stavu sa pokračuje s vytvorením novej matice kovariancie P_k z hodnôt

$$P_k = (I - KH)P_{k_p}. \tag{1.15}$$

Po vytvorení vypočítaných hodnôt sa iteruje od začiatku. Kalmanov filter používaný pri sledovaní objektu, ktorý bol použitý v [10] obsahoval X_k vektor priestoru obsahujúceho veľkosť detekcie, jeho pozíciu, rýchlosť zmeny pozície a rýchlosť zmeny veľkosti objektu. Pre dané hodnoty je vytvorený vektor, ktorého obsahom sú hodnoty potrebné pre aproximáciu. Potom je nutné vytvoriť matice A a H, ktoré vznikajú v závislosti od vektorov x_k a z_k. Kalmanov filter je v súčastnosti používaný pri sledovaní viacerých objektov vo videu. Jeho hlavnou vlastnosťou je invariancia voči oklúzii dvoch objektov, nakoľko pri zaniknutí detekcie vo videu predpokladá lineárny postup objektu[12].

1.1.4 Sledovanie objektu pomocou kontúr

Značí sledovanie objektu podľa špecifických častí objektu v obraze. Kontúrou objektu sa myslí špecifická časť objektu rozdielna od ostatných kontúr tak, aby ju bolo možné nájsť v inom obraze v ktorom sa objekt nachádza. Sledovanie je vykonávané na základe vyhľadávania počiatočných kontúr z predošlého obrázku. Objekt je reprezentovaný danými kontúrami[13]. Tieto kontúry môžu byť podľa potreby veľmi malé, pokiaľ je potreba zisťovať rozdiely medzi podobnými objektami, alebo veľké, ak je požiadavka klasifikovať objekty podľa ich podobnosti. Tak isto ako pri veľkosti je možné zväčšiť alebo zmenšiť počet kontúr objektu v závislosti od potreby konkretizovať objekt. Tu sa však musí dosiahnuť kompromis tak, aby bolo sledovanie invariantné voči zmenám objektu a aby nedochádzalo k výmene dvoch objektov. Cieľom detektorov spomenutých nižšie je vyhľadať tieto kontúry a správne ich popísať pre porovnanie. Sledovanie pomocou kontúr objektu je jednou z najpoužívanejších

pomôcok pre sledovanie objektu dodnes, a preto si ukážeme niektoré metódy používané v súčasnosti[14][15].

Škálovo invariantná transformácia vlastnosti

Škálovo invariantná transformácia vlastnosti(z angl. SIFT), podľa ktorej je možné vyhľadať použiteľné kontúry obrazu, ktoré sú následne algoritmicky opísané deskriptorom pre porovnanie s nasledujúcim obrazom. Pomocou tohto princípu sa vyhľadajú jednotlivé kontúry v obraze a priraďujú sa k danému objektu. Hlavným princípom SIFTu je použitie rôznych škál a parametrov masky, ktorým docieli odozvu rôznych veľkostí kontúr. Škálovým parametrom používaným v SIFT je hodnota σ, ktorá je parametrom Gaussovho filtra. Druhým je veľkosť masky docielená zmenšovaním veľkosti obrazu. Tieto dva parametre sa zisťujú pomocou grafu nulových kontúr [16], z ktorých je možné vyčítať nejlepšie parametre. V praxi sa používa parameter $\sigma = 1.6$ a mocniteľ tohto parametru $k = \sqrt{2}$. Zmena veľkosti obrazu prebieha pomocou DOG Gaussovej redukcie [17]

$$\sigma\Delta^2 G(k-1) \approx G(x, y, k\sigma) - G(x, y, \sigma), \tag{1.16}$$

kde $G(x, y, k\delta)$ a $G(x, y, \delta)$ sú stupne Gausovho filtru podľa násobku k. Extrémy takto vytvorených škál potom porovnávame s vyšším a nižším stupňom tak, aby sa zaistila najlepšia odozva. Na dané extrémy je použitá Hessova matica[18]

$$H = \begin{bmatrix} \frac{\partial^2 D}{\partial x \partial y} & \frac{\partial^2 D}{\partial y^2} \\ \frac{\partial^2 D}{\partial x^2} & \frac{\partial^2 D}{\partial x \partial y} \end{bmatrix} \tag{1.17}$$

ktorou sa ukáže, či je daný extrém kontúrou použiteľnou pre opísanie objektu. Túto maticu potom je treba upraviť tak, aby výsledkom bolo iba jedno číslo, ktoré je jednoduchšie porovnávateľné. Príkladom je Szeliskiho detektor[19]

$$R = \frac{STOPA(H)}{Det(H)}. \tag{1.18}$$

Po vyhľadaní potrebných oblastí sa postupuje s jeho opisom pomocou deskriptoru. Deskriptor používa oblasť okolo extrému, to je napríklad štvorec. Vytvorí centrálne derivácie v x-ovom a y-ovom smere, z ktorých vytvorí smer a veľkosť bodu podľa[20]

$$m(x, y) = \sqrt{(L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1)^2)}, \tag{1.19}$$

$$\theta(x, y) = \arctan\left(\frac{(L(x, y+1) - L(x, y-1)}{L(x+1, y) - L(x-1, y))}\right). \tag{1.20}$$

kde $m(x, y)$ je výpočet odvesny z derivácii v x-ovom a y-ovom smere. $\theta(x, y)$ je výpočet smeru derivácii okolo bodu $L(x, y)$. Jednotlivé veľkosti a smery sa zlúčia do celkov, pričom smery sa kvantizujú do ôsmich smerov, aby bol výpočet jednoduchší.

Časticový filter

Je vzorkovacia technika Bayesovskej sekvenčnej dôležitosti, ktorá rekurzívne aproximuje nasledujúcu distribúciu pomocou konečného počtu vážených vzoriek[21]. Tieto vzorky nazývané častice sú náhodne rozprestrené v obraze pričom zameriavajú rôzne vlastnosti a pomocou podobnosti objektu vytvárajú funkciu hustoty pravdepodobnosti. V stavovom popise systému tieto častice aproximujú teoretickú distribúcu objektu. Stavový popis systému sa skladá z procesu pozorovania $p(Z_{1:t}|X_t)$ kde Z predstavuje vektor pozorovania a X reprezentuje stavový vektor v čase t. Druhý proces stavového popisu je prechodový $p(X_t|X_{t-1})$ kde X je stavový vektor v časoch t a t-1. Systém časticového sledovania používa pravdepodobnostný systém používajúci prechodový proces pomocou všetkých pozorovaní z predchádzajúcich stavov $Z_{1:t-1} = Z_1, ..., Z_{t-1}$ na predikciu

$$p(X_t|Z_{1:t-1}) = \int p(X_t|X_{t-1})p(x_{t-1}|Z_{1:t-1})dX_{t-1} \tag{1.21}$$

Následne v čase t, kedy je možné zistiť Z_t je zistený stavový popis pomocou Bayesovho pravidla

$$p(X_t|Z_{1:t}) = \frac{p(Z_t|X_t)p(X_t|Z_{1:t-1})}{p(Z_t|Z_{1:t-1})} \tag{1.22}$$

kde model $p(Z_t|X_t)$ je získaný z procesu pozorovania. Následne je model v ďalšom procese aproximovaný pomocou váh w_t^i pre každú časticu i. Váhy vzoriek sú:

$$w_t^i = w_{t-1}^i \frac{p(Z_t|\tilde{X}_t^i)p(\tilde{X}_t^i|X_{t-1}^i)}{q(\tilde{X}_t|X_{1:t}, Z_{1:t})}, \tag{1.23}$$

kde $q(\tilde{X}_t|X_{1:t}, Z_{1:t})$ je distribúcia dôležitosti váh vybraných kandidátov, čiže vzoriek $\tilde{X}_t^i$. Častice sú následne prevzorkované aby bolo zabránené degenerácii dôležitosti častíc.

1.1.5 Sledovanie pomocou textúr

Textúra je miera zmeny intenzity povrchu, ktorá kvantifikuje vlastnosti ako hladkosť a pravidelnosť. Pomocou pravidelnosti môžeme sledovať objekty, ktoré sú reprezentované pomocou lokálneho dvojitého vzoru v [22]. Tento typ vzoru je invariantný voči zmene intanzity svetla, pričom používa čiernobiele textúry v lokálnych častiach obrazu. Taktiež je výhoda tohto typu sledovania v jeho jednoduchosti, aj keď existujú jeho zložitejšie varianty ako lokálne trojité vzory alebo viacúrovňové binárne vzory. Do popredia sa dostaly Gáborove vlnky s reprezentáciou a detekciou tváre pomocou Gáborových binárnych vzorov v [23]. Tieto typy reprezentácie sú používané pre rozpoznávanie objektov, avšak Gáborove vlnky je možno použiť aj pre sledovacie systémy, a preto je popísaná v nasledujúcej časti.

Gáborove vlnky

Je dvojrozmerná funkcia ktorá je používaná k detekcií frekvencí v rôznych smeroch. Jej funkcia je definovaná ako:

$$g(x,y) = (\frac{1}{2\pi\sigma_x\sigma_y}) \exp[-\frac{1}{2}(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}) + 2\pi jWx] \tag{1.24}$$

, kde σ_x a σ_y sú štandardné odchýlky v x-ovej a y-ovej osi. W predstavuje frekvencie záujmu. Následne je Gáborova vlnka vytvorená zmenou škály a rotáciou tejto funkcie

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = a^{-m} \begin{bmatrix} \cos\theta_n & \sin\theta \\ -\sin\theta_n & \cos\theta_n \end{bmatrix} \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \end{bmatrix}, \tag{1.25}$$

pričom na ľavej strane rovnice sa nachádza Gáborova vlnka g_{mn}, a je škálovací faktor pre ktorý platí $a > 1$. Škála sa nastavuje pomocou parametru $m = 1 : M$. Orientácia Gáborovej vlnky je určená podľa $\theta_n = n\pi/N$, pričom $n = 1 : N$. Konvolúciou obrazu s Gáborovou vlnkou získavame vlnkovú transformáciu:

$$\hat{I}(x,y,m,n) = \int I(x',y')g_{m,n}(x-x',y-y')dx'dy'. \tag{1.26}$$

Z tejto rovnice vzniká obraz, pričom pre každý vzniká M frekvenčných koeficientov s N počtom orientácii. Pokiaľ vypočítame Gáborovu vlnkovú transformáciu potrebujeme vybrať body, ktoré čo najlepšie reprezentujú objekt. Nakoľko koeficienty GWT sú počítané aj z oblati pixelu, nebude potrebné vybrať všetky body. Taktiež sa používa pri výbere správnych bodov prahovanie, aby sa zabrániu vplyvu šumu na sledovanie. Ďalšou možnosťou výberu bodov v lokálnom susedstve je výber bodu s najvyšším gradientom korešpondujúce s hranami a rohmi objektu. Body s najvyšším lokálnym maximom sú vybrané ako črty objektu[24]. Po vyhľadaní týchto bodov je potreba použiť vzťahy medzi týmito bodmi na sledovanie, a to pomocou sieťe. Z N_p počtu vybraných bodov vyberieme nový bod z $(x_f(i), y_f(i))$ z N_p a pripojíme ho pomocou líniového segmentu k predchádzajúcim bodom $(x_f(j), y_f(j))|j = 1 : i-1$. Ak sa pridaný segment kríži s iným, vyberieme kratší z nich. Pre reprezentáciu sieťe je potrebné aby bol invariantný voči zmene afinity a zmeny veľkosti. Sieť je reprezentovaná pomocou vektorov trojuholníkov v priestore, ktoré sú vytvorené pomocou líniových segmentov $G = [a(1), a(2), ..., a(N_t)]$, kde a(i) je rozloha i-teho trojuholníku. Transformácia afinity pre body je vytvorená:

$$\begin{bmatrix} x_a \\ y_a \end{bmatrix} = C \begin{bmatrix} \mathrm{x} \\ \mathrm{y} \end{bmatrix} + \begin{bmatrix} \mathrm{g} \\ \mathrm{h} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix} + \begin{bmatrix} \mathrm{g} \\ \mathrm{h} \end{bmatrix}, \tag{1.27}$$

pričom x a y je originálna pozícia bodu. Vo vzorci sú x_a a y_a body po zmene afinity. Po vytvorení modelu je treba vypočítať podobnosť medzi sledovanými objektami a

to pomocou súvislosťami medzi bodmi, líniami a trojuholníkmi. Konečná podobnosť bude váženým súhrnom jednotlivých podobností. Pre výpočet lokálnej podobnosti je treba vytvoriť súhrn jednotlivých podobností medzi časťami:

$$S_l = \sum s_t(i)_{i=1}^{N_p}/N_p, \tag{1.28}$$

kde jednotlivé individuálne lokálne podobnosti počítame ako:

$$s_l(i) = \frac{L_O(i).L_M(i)}{||L_O(i)|| \times ||L_M(i)||} \min(\frac{||L_M(i)||}{||L_O(i)||}, \frac{||L_O(i)||}{||L_M(i)||}), \tag{1.29}$$

pričom $L_M(i)$ je lokálny vektor detekcie a $L_O(i)$ je lokálny vektor sledovaného objektu. Druhou funkciou podobnosti ktorá je počítaná je globálna, ktorú počítame ako:

$$S_g = \cos(k \times \arccos(\frac{G_M.G_O}{||G_M|| \times ||G_O||})), \tag{1.30}$$

kde G_M je vektor obsahu detekcie, G_O je vektor obsahu objektu a parameter k je určený k zmene citlivosti na deformáciu[24]. Výsledná podobnosť objektu a detekcie vzniká zlúčením globálnej a lokálnej podobnosti a výberom parametru α:

$$S = \alpha S_t + (1 - \alpha)S_g. \tag{1.31}$$

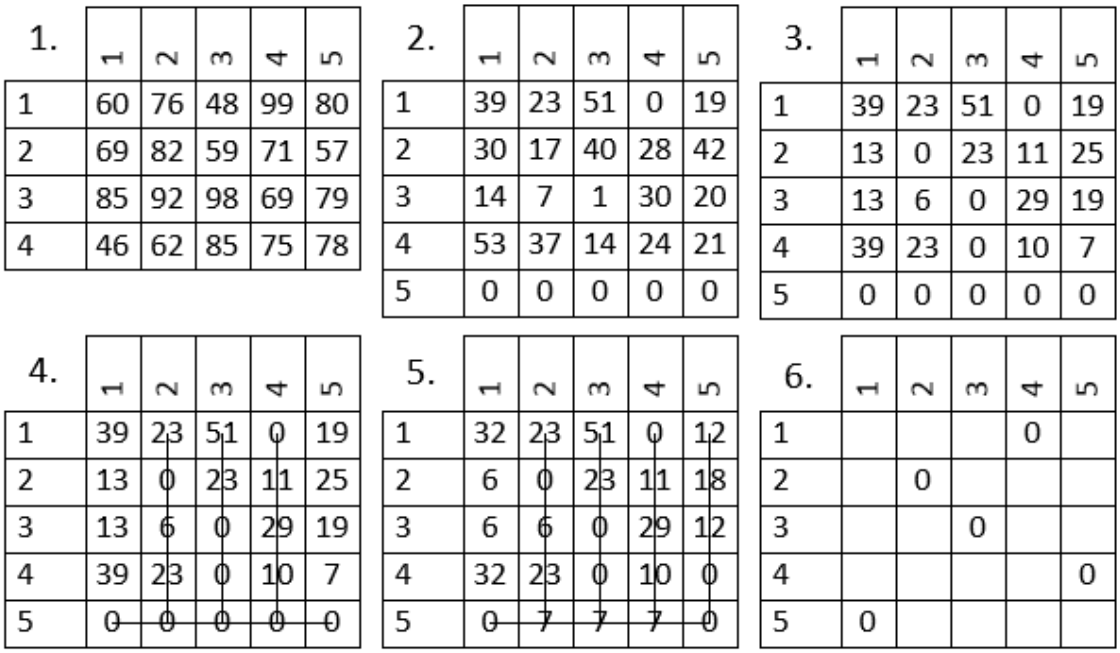
Po porovnaní detekcie s objektom sa pokračuje s vyhľadaním objektu v obraze. Používanými metódami sú vyhľadanie miesta objektu vo všetkých možných polohách alebo vyhľadanie jednotlivých bodov alebo línii v obraze.

1.2 Problém priradenia objektu

Je jeden z fundamentálnych optimalizačných problémov v matematike, ktorý rieši priradenie maximálnou celkovou váhou vo váženom biparitnom grafe. Problém priradenia objektu sa používa pri sledovaní viacerých objektov, pričom rieši priradenie detekcie k objektu. Tento algoritmus rozhoduje o tom, či detekcia reprezentuje objekt viac, ako súbor iných detekcií[25]. Tento typ problému môžeme vyobraziť ako biparitný graf reprezentovaný ako list trojíc *y,z,l,* pričom každý y reprezentuje objekt, z predstavuje detektor [26]. Výsledok algoritmu by mal vytvoriť dvojice $y_i, z_i,$. Príkladom algoritmu na riešenie tohto problému je Maďarská metóda priraďovania[27], ktorá rieši daný problém v polynomiálnom čase. Táto metóda používa na vyriešenie problému priradenia maticu, ktorú je možné jednoducho vytvoriť z biparitného grafu tak, že objekty reprezentujú riadky a stĺpce detekcie, pričom v jednotlivých bunkách sú váhy medzi nimi[25].

1.2.1 Maďarská metóda priraďovania

V tejto sekcii je popísaný postup priraďovania Maďarskou metódou[27]. Prvým krokom je pozorovanie, či je rovnaký počet riadkov a stĺpcov matice. Príčinou tejto skutočnosti je fakt, že detektor nemusí detegovať objekt alebo možnosť, že do obrazu pribudne nový objekt. Ak takýto jav nastane, je potreba odčítať od každého prvku najmenší prvok matice a vytvoriť další riadok tak, aby bol počet riadkov a stĺpcov rovnaký. Druhým krokom metódy je odčítavanie najmenšieho prvku v riadku od každého prvku v riadku. Rovnaký postup je použitý aj pre každý stĺpec. Týmto postupom sa získajú nulové hodnoty v matici. Pokiaľ pri opakovanom prechode maticou v riadkoch príde na nulu, je potrebné vytvoriť vertikálnu čiaru zakrývajúcu všetky hodnoty v tomto smere. Táto operácia sa opakuje aj pri prechádzaní v stĺpcoch, avšak pri zistení nuly sa vytvára diagonálna čiara. Pokiaľ sa počet vytvorených čiar rovná počtu riadkov, je možné vytvoriť optimálne priradenie. Nezačiarknuté nulové hodnoty, sú vyjadrením priradenia detekcie ku objektu. Ak sa však počet vytvorených čiar nerovná počtu riadkov, je treba vytvoriť novú maticu a to tak, že sa vyhľadá najmenší prvok zo súboru neoznačených čísel, ktorý je následne odčítaný od jednotlivých čísel z tohto súboru. Potom je potrebné vyhľadať prvky, ktorými prechádzajú čiary vo vertikálnom aj diagonálnom smere, ku ktorým je pripočítaný najmenší prvok z predchádzajúceho kroku. Pokiaľ je vytvorená nová matica, je možné opakovať proces od začiatku. Výsledné matice po jednotlivých krokoch je možné vidieť na obr.1.1



Obr. 1.1: Postup získania optimálneho priradenia pomocou Maďarskej metódy priradenia

1.2.2 Enumeračná metóda priraďovania

Je najjednoduchším riešením priraďovacieho problému. Jeho hlavnou myšlienou je použitie hrubej sily pre daný problém, a to výpočtom všetkých možností priradenia[28]. To spôsobuje pomalý výpočtový výkon, nakoľko počet možných priradení je faktoriál veľkosti matice. V prípade matice 3x3 je veľkosť výpočtov možností až 3!.

1.3 Zlučovanie metód sledovania

Táto časť sa zaoberá metódami pre zlučovanie výsledkov sledovacích systémov do jedného celku. Reprezentácia objektu prináša zlepšenie pri sledovaní objektu za cenu vyššej výpočtovej zložitosti. Použitie viacerých metód pre opis objektu taktiež zvyšuje robustnosť voči chybám, vyskytujúcim sa pri použitý jednotlivých metód samostatne. Zlučovacie systémy je možné deliť pomocou použitia rôznych modelov ohodnotenia jednotlivých systémov, adaptívnosti systému a použitia paralelného alebo kaskádovitého systému hodnotenia.

1.3.1 Paralelné a kaskádovité hodnotenie sledovacích systémov

Pri sledovaní rozlišujeme viacero možností ako zlúčime sledovacie systémy. V paralelnom systéme v každom kroku je vybraný najspoľahlivejší kandidát pre sledovanie, prípadne váhovanie jednotlivých metód do konečného výsledku. V kaskádovitom systéme je vyhodnotené sledovanie postupne, a ak sa dosiahne vysokej percentuálnej istoty, že ide o správny výsledok, je vybraná táto možnosť[29]. Kaskádovitý systém je pre svoju postupnosť procese nenáročný, avšak je potrebné aby tu existovalo správne poradie jednotlivých sledovacích systémov, ako aj správny prah kedy systém zaradí výsledok za správny. Poradie jednotlivých sledovacích systémov je zoradené podľa ich váhy, čiže miery dôvery v tento sledovací systém.

1.3.2 Vážené ohodnotenie vlastností

Vážené ohodnotenie vlastností [30] vytvára status objektu ako jeho pozíciu, výšku a dĺžku objektu. Tu je ohodnotenie objektu tvorené Haarovými funkciami[31], intenzitou obrazu a jeho histogramom. Spočiatku je vytvorený iniciálny stav pomocou Harrisovho detektora rohov[32]. Následne sú vytvorené vzorky kandidátov okolo iniciálneho stavu váženého podľa Gausovej distribúcie. Potom sú ohodnotený jednotlivý kandidáti podľa vlastností zvlášť a následne sú zlúčený pomocou váženej entropie. Translácia objektu v obraze je v nasledujúcom obraze riešená pomocou Harrisových rohov. Najlepšie reprezentácie objektu sú vytvorené pomocou podobrázkov, pričom podobrázok s najnižšou Euklidovou vzdialenosťou od modelu vzhľadu je vybraný ako bázový. Najprv sú tieto rohy detegované a následne jednotlivo vyhľadané translácie v nasledujúcom obraze. Výsledná translácia objektu je vypočítaná ako priemer translácii jednotlivých rohov. Po tomto kroku sa znova môže pokračovať v ohodnotení pomocou viacerých vlastností.

1.3.3 Diskriminatívne ohodnotenie vlastností

Je vážená entropia, ktorá diskriminatívne kombinuje vlastnosti pre sledovanie [30]. Tu existuje N možných stavov, ktoré sú hodnotené podľa intenzity obrazu L^{inten}, histogramu L^{hist} a Haarových funkcii L^{haar} a časovom okamihu t. Hodnotenia sú definované ako:

$$p(O_t|X_t) = \exp(-c||(F_t - \bar{F}_t) - UU^T(F_t - \bar{F}_t)||_2^2), \tag{1.32}$$

pričom U je analýzou hlavných komponentov podpriestorov, F_t je vlastnosť objektu a $\bar{F}_t$ je zmena od predošlej vlastnosti. Vytvorením pravdepodobnosti p z pozorovania

O_t v čase t, vlastnosti F_t, je určená pravdepodobnosť stavu X_t. Po vytvorení pravdepodobností je zachovaná uniformnosť ohodnotenia jeho normalizovaním[2]. Ohodnotenie s najnižšou váhou je vymazané. Po normalizácii sumy hodnotení pravdepodobnosť i-teho kandidáta bude:

$$p_i = \alpha_0 \tilde{L}_i^{inten} + \alpha_1 \tilde{L}_i^{hist} + (1 - \alpha_0 - \alpha_1)\tilde{L}_i^{haar}, \tag{1.33}$$

kde α_0,α_1 a $1 - \alpha_0 - \alpha_1$ sú koeficienty jednotlivých ohodnotení. $\tilde{L}_i^{inten}$, $\tilde{L}_i^{haar}$ a $\tilde{L}_i^{hist}$ sú normalizované ohodnotenia i-teho kandidátskeho stavu. Vyhodnotenie pravdepodobnosti jednotlivých kandidátov, ktorý sú následne porovnaný pre výber najpravdepodobnejšieho kandidáta.

1.3.4 Adaptívnosť sledovacieho systému

Je možnosť sledovacieho systému prispôsobiť sledovací systém pre zníženie jeho chybovosti. Adaptívnosť pri zlučovaní je používaná pri zmene váh, alebo typu zlučovacej stratégie. V tejto časti si popíšeme jeden z typov adaptívnosti. V sledovacom systéme popísanom v [34] je adaptívnosť dosiahnutá v zmene typu zlučovania. Model zlučovania súčinovým pravidlom predpokladá, že jednotlivé vlastnosti sú od seba nezávislé. Potom je výsledná pravdepodobnosť stavu x:

$$p(z^1 ... z^n|x) = \prod_{i=1}^{n} p(z^i|x), \tag{1.34}$$

kde z^i je ohodnotenie i-tej vlastnosti odhadovaného stavu x. Použitie tohto typu zlúčenia napomáha presnosti sledovania, ktoré má vysokú kvalitu obrazu pretože jednotlivé ohodnotenia vlastností ukazujú relatívne rovnaké výsledky. Druhá možnosť zlúčenia je model váženej sumy:

$$p(z^1 ... z^2|x) = \sum_{i=1}^{n} \alpha^i p(z^i|x), \tag{1.35}$$

pričom a^i sú váhy i-tej vlastnosti, ktoré sú normalizované tak aby ich súčet bol jedna. Tento systém zlučovania je používaný, pokiaľ je kvalita obrazu nízka, respektíve pokiaľ jednotlivé ohodnotenia vlastností so sebou nesúhlasia. Potom podľa koeficientu určujúceho či jednotlivé ohodnotenia so sebou súhlasia určíme ktorý z modelov použijeme.

1.4 Problémy nastávajúce pri sledovaní objektu

Táto sekcia pozostáva z popisu rôznych typov problémov, ktoré môžu nastať pri sledovaní objektu a ich riešenie pomocou používaných techník v dnešnej dobe, nakoľko vyššie uvedené sledovacie systémy sú náchylné na chyby.

[2]Normalizovaný vektor $\hat{\mathbf{X}}$-je vektor s dĺžkou 1 v rovnakom smere ako vektor $\mathbf{X}$ [33]

1.4.1 Sledovanie častí, ktoré nie sú objektom

Tento problém nastáva pri zobrazovaní detekcie pomocou primitívnych objektov, ako napríklad obdĺžnik alebo elipsa. Určitá časť reprezentácie objektu zahŕňa aj jeho okolie, ktoré žiadnym spôsobom nepopisuje objekt. Popis okolia objektu spôsobuje problémy hlavne pri sledovaní pomocou farieb detekcie alebo kontúr. Pri sledovaní pomocou farieb sa vytvorí histogram nielen na základe objektu, ale aj na základe okolia objektu a pri posunutí objektu môže nastať priradenie detekcií k nesprávnemu objektu nachádzajúcemu sa v rovnakej oblasti. Taktiež pri sledovaní pomocou kontúr, zvlášť ak má okolie dobré kontúry pre porovnanie, môže vzniknúť zámena priradenia k objektom. Tento problém je riešený pomocou odstránenia pozadia fungujúceho na báze Canny detektoru hrán[4], ktorý je popísaný nižšie.

Canny detektor hrán

Canny detektor hrán je jedným z najpoužívanejších detektorov hrán v súčasnosti. Využíva derivačný Gaussov filter, zužovanie hrán a hysterézne prahovanie, ktoré si popíšeme z [35]. Gaussova derivačná filtrácia znamená derivácia Gaussovho filtru v dvoch smeroch, pri ktorých vzniknú dve masky. Konvolúciou s obrázkom potom dostaneme dva výsledky, ktoré spojíme

$$S_{xy} = \sqrt{S_x^2 + S_y^2}, \tag{1.36}$$

kde S_x a S_y sú gradienty filtrácie na osi x a y. Zužovanie hrán je erózia hrany, pričom zostáva z hrany iba jej maximálna zložka. Funguje na princípe lokálneho maxima v smere hrany, ktoré sa dá jednoducho zistiť pomocou

$$\Theta = tan^{-1}\frac{S_y}{S_x}. \tag{1.37}$$

Lokálne maximum potom určité detekcie ponechá a všetky ostatné zruší. Po zužovaní hrán sa postupuje v prahovaní pomocou dvoch prahov, a to horného, ktorý automaticky zaradí všetky hrany nad tento prah a dolného, ktorý vylúči všetky hodnoty pod ním. Stred je určovaný podľa toho, či hrana pokračuje nad spodným prahom. Výslednú detekciu hrán je možno vidieť na obr.1.2.

a) b)

Obr. 1.2: a)Spracovávaný obraz b)Použitie detektoru hrán Canny

Po vytvorení hrán sa postupuje nastavením ukotvenia obrázku, pričom časti oddelené hranami sú odstránené [36].

1.4.2 Šum obrazu

Šum obrazu je problém, ktorý sa stáva menšou prekážkou vzhľadom na zlepšujúcu sa kvalitu obrazov, avšak môže zavážiť hlavne pri sledovaní pomocou menších kontúr, ktoré majú veľkú náchylnosť na chyby tohto typu. Vznik tohto typu problému závisí od zlej kvality alebo nastavení kamery. V súčasnosti je používaná filtrácia proti danému typu problému alebo použitie metódy sledovania, ktorá je invariantná voči šumu obrazu. Jedným z jednoduchších filtrov proti tomuto typu je mediánový filter. Jeho základným princípom je výpočet mediánu z určitej veľkosti filtru

$$f(x,y) = med(f(x-1,y-1), f(x,y-1), ..., f(x+1,y+1)). \tag{1.38}$$

kde sú $f(x-1,y-1), f(x,y-1), ..., f(x+1,y+1)$ hodnoty z okolia bodu $f(x,y)$.

1.4.3 Zmena osvetlenia obrazu

Zmena osvetlenia obrazu je problém vyskytujúci sa pri vysokom osvetlení alebo nelinearite osvetlenia objektu. Problém tohto typu sa vyskytuje prevažne pri jednoduchých detekčných metódach, ako je subtrakcia pozadia alebo porovnávanie histogramu. Pri lineárnom osvetlení je uvedený obraz ako priestor, pričom jednotlivé jeho časti sú rozdelené do podpriestorov, kde každý podpriestor má iné vlastnosti prostredia [37]. Problém pri porovnávaní histogramu je možné riešiť na základe porovnávania diferencii medzi jednotlivými veľkosťami histogramu na rozdiel od porovnávania veľkosti stupňov jasu ako takých. Avšak aj tu nastáva problém pre nelineárne osvetlenie objektu. Hlavnou metódou riešenia daných problémov je použitie

detektoru invariantného voči daným problémom, a to je použitie detektoru kontúr, konkrétnejšie bodov záujmu.

1.4.4 Zjavenie a zánik objektu v obraze

Táto časť sa zaoberá otázkou spôsobu vyhľadania objektu v obraze, čím je vytvorený jeho zjav. Tento problém sa rieši v súčasnosti pomocou súboru vlastností daného objektu. Prvé riešenie pomocou šablón, príkladov objektu v obraze, môžeme vytvoriť z malého počtu vzorov, a tým zaistiť rýchle spracovanie údajov za cenu potreby tuhého objektu. Pokiaľ je potreba vytvoriť šablóny objektu pre komplexnejšie objekty, je treba použiť väčší počet šablón, čím sa zvýši výpočtová zložitosť. Druhým typom riešenia je použitie súboru vlastností, ktorými sa zaoberajú Haarove vlastnosti alebo SIFT kontúry. Problémom týchto detektorov je však potreba zachovať určitú štruktúru objektu pre jeho kategorizáciu. Sledovanie pomocou histogramu je jednou z metód invariatných voči radikálnej zmene otočenia alebo zmeny tvaru objektu. Preto sa v dnešnej dobe používajú histogramy orientovaných gradientov ako riešenie zjavu chodcov[38].

1.4.5 Prekrytie objektu

Prekrytie objektu tiež nazývané oklúzia je stav, kedy kľúčové atribúty pre detekciu objektu nie sú k dispozícii pre kameru z dôvodu jeho polohy v obraze. Vznik oklúzie môže nastať rôzne, napríklad prekrytím s objektom v pozadí, prekrytím s iným objektom, ktorý detekujeme alebo prekrytím inou časťou detekovaného objektu. Neschopnosť detegovať objekty, ktoré medzi sebou prechádzajú, má za následok zlyhanie detektoru, hlavne pri reprezentácii pomocou kernelu [39][40]. Riešenie problému oklúzie znamená vedieť priradiť objekt aj po prekrytí a stratení detekcie. Tento problém môže zhoršiť aj možnosť, pri ktorej majú prekrývajúce sa objekty vysokú mieru podobnosti, t.j. pri ktorých sú detekované rovnaké

- farby
- kontúry
- veľkosti

Prekrývanie objektov je možné rozdeliť podľa veľkosti prekrytej časti do troch skupín. Prvou skupinou sú objekty v obraze bez oklúzie, ktorého všetky atribúty môžeme sledovať počas celého priebehu sledovania. Ďalej sú to objekty, ktoré sú prekryté iba časťou, takže existuje menej individuálnych znakov, ktoré je možné sledovať, a preto sa stáva sledovanie náročnejším. Avšak tu sa naskytá možnosť použitia sledovacích metód, ktoré neberú do úvahy tvar objektov, napr. sledovanie pomocou Mean Shift vektoru spomenutého v 1.1.1. Existuje tiež metóda vytvorenia hĺbky scény na základe hustoty pravdepodobnosti z trénovacích obrázkov[41]. Posledným

typom podľa veľkosti prekrytej časti objektu je úplná oklúzia, pri ktorej neexistuje žiadna možnosť detegovať objekt. Tu je možné použiť metódy predikcie polohy objektu na základe lineárneho alebo nelineáreho modelu pohybu objektu. Jedným z príkladov lineárnej predikcie pohybu je vyššie spomenutý Kalmanov filter.

1.5 Zhrnutie

V tejto časti sú v jednoduchosti zhrnuté použiadavky na systémy sledujúce viaceré objekty.

Pre úspešné sledovanie viacerých objektov v obraze je nutné aby bola zaručená robustnosť systému. Tým sa myslí zaručenie sledovania pri ťažkostiach nastávajúcich pri sledovaní, ktoré boli spomenuté v časti 1.4.5.

Druhou nutnou požiadavkou na systém je práca v reálnom čase, čo je systém ktorý vie spracovať minimálne 15 snímkov za sekundu, aby sa zaistil plynulý výstup pre ľudské oko[42]. Poslednou vlastnosťou systému je možnosť prispôsobiť sa rozdielnym prostrediam, ktoré sú sledované alebo prispôsobiť systém pre rôzne typy objektov podľa potreby. Veľký rozsah prác bol z hľadiska robustnosti vytvorený, pričom niektoré z nich sú spomenuté v časti 1.1. Jedným z nich je napríklad Kalmanov filter[43], ktorý predikuje postup objektu na základe jeho predchádzajúceho postupu. Tento postup je lineárny, čo je nepriaznivé pre sledovanie pri abruptnej zmene. Kalmanov filter je často používaný pri sledovaní v reálnom čase, avšak ak sa zvýši počet sledovaných objektov dochádza k zvýšenej chybovosti. Ďalším systémom používaným pri sledovaní je Mean-shift, ktorý predpokládá malú rýchlosť pohybu objektu, čo ho z časti limituje. Taktiež nedokáže čeliť úplnému prekrytiu objektov. Tento systém je prevažne používaný pri sledovaní stredného počtu objektov a často sa používa v kombinácii s Kalmanovým filtrom pre riešenie oklúzii[44].

Problémy pri vyššie spomenutých systémoch rieši časticový filter, ktorý doteraz nebol používaný, vzhľadom na jeho výpočtové nároky. So zlepšujúcou sa technikou sa stal tento systém populárny avšak čím vyššiu presnosť sledovania je potrebné dosiahnuť, tým väčšiu výpočtovú zložitosť má tento systém. Taktiež tento systém čelí degenerácii váh častíc čím spracuváva údaje ktoré nie sú relevantné pre správne určenie polohy objektu v obraze. Preto je pre používanie Časticových filtrov potrebné opatrné vyladenie parametrov, čím sa znižuje prispôsobivosť systému na rozličné prostredia. V súčastnej dobe sa používa časticový filter v spojení s Markovým reťazcom Monte Carlo. V tejto práci je narhnutý systém zlučujúci viaceré vlastnosti ktoré používame pri sledovaní.

Rovnakým spôsobom konfrontuje problém sledovania pravdepodobnostné zlúčenie senzorov v [45]. Týmto systémom sledovania je zaistená robustnosť voči oklúzii, zlyhaniu senzorov a iných chýb spomenutých v časti 1.4.5. Zlúčenie jednotlivých

systémov bolo v práci [46] vytvorené pravdepodobnostne, alebo paralelne a kaskádovito [29]. Taktiež je potrebná adaptívnosť systému, ktorá je dosiahnutá v práci zmenou váh jednotlivých typov sledovania.

2 PRAKTICKÁ ČASŤ

2.1 Nástroje použité pre tvorbu

Táto časť zahŕňa výber prostredia a nástrojov použitých v práci a dôvody ich použitia.

2.1.1 Eclipse Jee Oxygen

Ako vývojové prostredie je používaný Eclipse Jee Oxygen, ktorý je open source. Eclipse je vhodný na tvorbu v programovacom jazyku Java, avšak jeho flexibilita umožnuje doplniť vývojové prostredie o programovacie jazyky ako C++ alebo PHP. Hlavnou výhodou tohto vývojového prostredia je možnosť rozšírenia pomocou pluginov, čím výrazne znižuje jeho veľkosť v pamäti.[47]

2.1.2 OpenCv 2.4.13

Open source knižnica OpenCv je hlavným prostriedkom pre obrazové spracovanie, ktorá je použitá v práci. Táto knižnica sa venuje počítačovému videniu v reálnom čase, čo je dôvodom pre výber tejto knižnice. OpenCv bola založená v roku 2000 spoločnosťou Intel pre podporu počítačového videnia v komerčnej sfére. Aplikácia OpenCv je veľmi robustná, nakoľko obsahuje viac ako 2500 algoritmov, či už pre detekciu objektov, v robotike, segmentácii a na sledovanie objektov. Taktiež jednou z výhod je veľký počet používateľov tejto knižnice presahujúci 47 tisíc. OpenCv podporuje programovacie jazyky ako C++, C, Python, Java pre operačné systémy Windows, Mac OS a Linux[48].

2.2 Programové riešenie

V tejto časti je popísaný postup pri tvorbe detekcie, riešenie hlavných problémov pri sledovaní objektu, použitie metód pre sledovanie a jeho zlúčenie do jedného celku.

2.2.1 Návrh sledovania

Je základom pri tvorbe sledovacieho programu. V tejto práci je sledovanie objektov demonštrované na príklade detekcie tvárií a chodcov. S týmto problémom úzko súvisí otázka, ako reprezentovať sledovaný objekt. Na výber sa ponúka niekoľko možností, a to reprezentovanie pomocou siluety, jednoduchých geometrických objektov,bodov alebo skeletálneho modelu. Sledovanie pomocou siluety sa používa pre subtrakciu pozadia, ktorá nie je použitá pre detekciu v tejto práci. Skeletálny model je možné

použiť pri mimike tváre, ktorá pri tomto type sledovania nie je využívaná a sledovanie pomocou bodov sa používa pre veľmi malé objekty. Rozhodnutie pripadá na reprezentáciu pomocou geometrického obrazca, konkrétnejšie obdĺžnika.

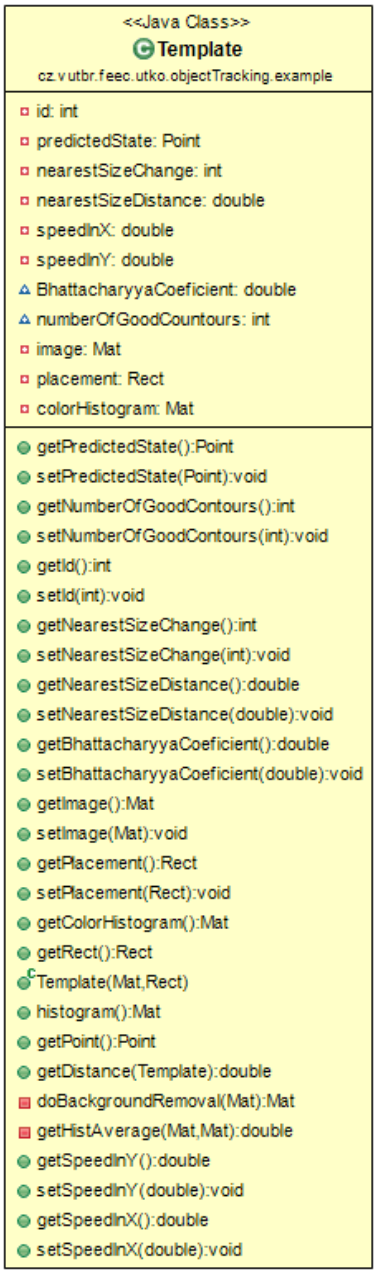
Po výbere reprezentácie objektu je treba preň vytvoriť model zjavu, čo znamená vlastnosti, pomocou ktorej reprezentujeme jednotlivé detekcie. Pokiaľ je objekt detekovaný, postupuje sa s výberom rôznych vlastností, podľa ktorých je objekt porovanávaný s detekciami v nasledujúcom obraze. Cieľom tejto práce bolo použiť viacero vlastností, a preto sú vyberané metódy sledovania podľa farby, vzdialenosti a kontúr. Pokiaľ sú ustanovené všetky tieto pravidlá, je možné vytvoriť triedu, obsahujúcu všetky informácie o vzore, podľa ktorého sú porovnávané jednotlivé detekcie v nasledujúcom obraze [49].

2.2.2 Porovnávanie so vzorom

Trieda, podľa ktorej je porovnávaná detekcia so vzormi, obsahujúca informácie o pozícii reprezentácie v obraze. Z hodnoty obrazu a pozície štvorca, ktorý je reprezentáciou objektu v obraze, je možné vyňať túto časť obrazu pre porovnanie pomocou metódy `frame.submat(new Rect())` z knižnice OpenCv, ktorá vyberie z obrazu určitú časť, v našom prípade obdĺžnik triedy `Rect` z OpenCv. Základný výrez z obrazu postačuje na to, aby sa z neho získali potrebné dáta na porovnanie. Jednou z týchto metód je histogram, podľa ktorého sú porovnávané detekcie so vzormi. Ďalšími metódami použitými pre sledovanie pomocou porovnávania so vzorom sú:

- polohy výrezov
- veľkosť vzoru
- kontúry vzoru.
- rýchlosť a predikcia polohy vzoru

Tieto hodnoty spájajú detekcie k jednotlivým vzorom podľa daného typu detekcie. `Template` je možne ukázať na diagrame 2.1



Obr. 2.1: UML diagram triedy template

Po vytvorení triedy `Template` sa postupuje tvorbou súboru vzorov potrebných na porovnanie. Súbor vzorov je vytvorený ako `Arraylist`, čo je zoznam voliteľnej veľkosti pre rôzne typy tried. Taktiež je možné manipulovať s rozsahom tohto radu alebo ho triediť pomocou vybranej vlastnosti elementu alebo iterovať cez rad. Vytvorením radu vzorov v predošlom snímku a radu detekcií v nasledujúcom vznikol biparitný graf, kde je riešený problém priradenia pomocou detektorov, ktoré je opísané v nasledujúcej časti.

2.2.3 Metódy sledovania

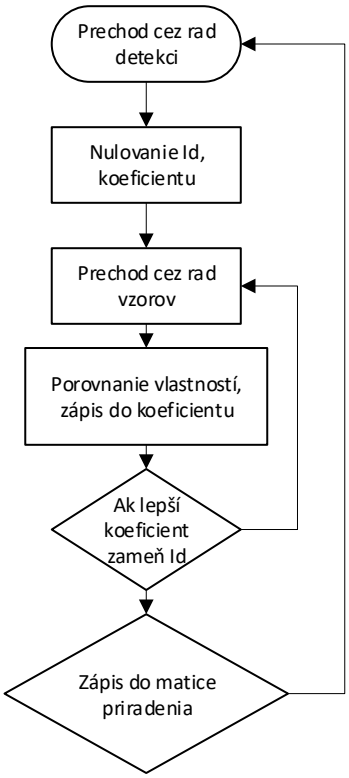
Metódy sledovania v praktickej časti obsahujú tvorbu rozhrania a algoritmus metód v triedach, ktoré tvoria sledovanie. Najskôr je potrebné zadefinovať rozhranie `IMatcher`:

```
public interface IMatcher {

  public void match(List<Template> listOfTemplates, List<
     Template> listOfDetections,double[][]
     assignmentArray);
  public void setMultiplicator(double multiplicator);
  public double getMultiplicator();


}
```

Všetky triedy implementujúce rozhranie `IMatcher` musia obsahovať metódu `match` a getter so settrom multiplikátora s použitými parametrami. Metóda `match` vyplní maticu priradenia pre podľa daného typu sledovania a `Multiplicator` je potrebný pri nastavení váhy jednotlivých typov sledovaní. Týmto spôsobom sú zaradené jednotlivé triedy, porovnávajúce vzorový rad s radom detekcií.

Metóda match

Metóda match je základnou metódou každého vytvoreného sledovacieho algoritmu. Skladá sa z dvoch iterácii cez rad detekcií a vnorenej iterácie prechádzajúcej radom vzorov. Tie porovnáva za pomoci vlastností nachádzajúcich sa v 1.1. Z týchto porovnaní je vytvorená hodnota alebo koeficient, ktorá určuje príbuznosť detekcie so vzorom. Pokiaľ zistila metóda sledovania najlepšiu zhodu, priradí danému detektoru index vzoru v rade zapísaním indexu do premennej `ID` podľa typu danej metódy. Tento algoritmus je možné vidieť na obr.2.2



Obr. 2.2: Algoritmus porovnávania vzoru s detekciou

Uvedený algoritmus prirovná detekcie k vzorom, pričom existuje možnosť prepísať vzor za prirovnanú detekciu alebo ponechať vzor. Prepísaním vzoru je docielená lepšia detekcia v nasledujúcom obraze, avšak pri chybe detekcie môže spôsobiť postup, teda kumulovanie chyby, a tým výrazne poškodiť sledovanie. Pokiaľ je vzor ponechaný, znižuje sa tým šanca dobrého sledovania, ale nenastane situácia postupu chyby pri sledovaní.

Porovnávanie pomocou farby

Porovnávanie pomocou farby je prvou z uvedených metód porovnávania, a taktiež jednou z metód použitých v tomto projekte. V skratke ide o porovnávanie histogramov, ktoré sú získané z metódy `histogram` triedy `template`:

```
public Mat histogram()
{
    Mat greyimage = new Mat(image.height(), image.width
       (), CvType.CV_8UC2);
```

```
        Imgproc.cvtColor(image, greyimage, Imgproc.
            COLOR_RGB2GRAY);
        Vector<Mat> bgr_planes = new Vector<Mat>();
        Core.split(greyimage, bgr_planes);
        MatOfInt histSize = new MatOfInt(256);
        final MatOfFloat histRange = new MatOfFloat(0f, 256
            f);
        boolean accumulate = false;
        Mat b_hist = new  Mat();
        Imgproc.calcHist(bgr_planes, new MatOfInt(0),new
            Mat(), b_hist, histSize, histRange, accumulate);
        return b_hist;
    }
```

Táto metóda vytvorí čiernobiely obraz z výrezu a zapíše ho do potrebného formátu `Vector<Mat>`. Čiernobiely model je použitý pre zjednotenie histogramu a pre zníženie výpočtovej náročnosti. Druhým krokom metódy je nastavenie údajov o histograme a následný zápis do histogramu pomocou metódy `calchist` z knižnice OpenCv. Pokiaľ sú vytvorené histogramy detekcie a vzoru, môže začať porovnávanie histogramov podľa Bhattacharyyovho koeficientu popísaného v časti 1.1.1. Pre výpočet koeficientu je použitý v programovacom jazyku metóda `compareHist` z OpenCv.

Porovnávanie pomocou vzdialenosti

Porovnávanie pomocou vzdialenosti je metódou sledovania, ktorá sa hodí prevažne na detekciu bez oklúzii. Pozoruje Euklidovu vzdialenosť jednotlivých detekcií voči vzorom, pričom predpokladá plynulý postup detekcie v obraze. Pri tvorbe sledovania je nutné vypočítať stredový bod objektu triedy `Template`:

```
public Point getPoint()
{
  Point templatePoint= new Point();
  templatePoint.x= (getRect().x+getRect().width)/2;
  templatePoint.y=(getRect().y+getRect().height)/2;
  return templatePoint;
}
```

Pokiaľ máme vytvorený stredový bod, je možné vypočítať Euklidovu vzdialenosť stredových bodov. Trieda vzoru obsahuje pre sledovanie vzdialeností metódu `getDistance`:

```
public double getDistance(Template otherTemp)
```

```
{
  double distance=Math.sqrt(Math.pow(getPoint().x-
     otherTemp.getPoint().x, 2)+Math.pow(getPoint().y-
     otherTemp.getPoint().y, 2));
  return distance;
}
```

Čím menšia je veľkosť Euklidovej vzdialenosti, tým je väčšia šanca, že je pár detekcia - vzor jedným objektom. Problém nastáva pri prechode detekovaných objektov za seba, čím má metóda sledovania porovnateľne veľké vzdialenosti jednej detekcie k vzorom dvoch objektov.

Párovanie Bodov záujmu

Je metóda používajúca body záujmu vzorov a detekcií, medzi ktorými vytvára korektné spojenia. Body záujmu tvoríme z objektu triedy `Template`, konkrétne z výrezu obrazu, ktorý táto trieda obsahuje. Pre vyhľadávanie bodu záujmov je použitá obmena SIFT a SURF[50], ORB[14] detektor bodov záujmu[14]. Po vyhľadaní bodov záujmu je nutné vytvoriť deskriptory, pomocou ktorých sú porovnávané jednotlivé body záujmu. Následne sú hrubou silou porovnávané jednotlivé deskriptory vzorov a obrazu čím vzniká priradenie bodu záujmu vzoru k bodom záujmu obrazu. Po zistení umiestnenia bodov záujmu detekcií v obraze algoritmus zisťuje, v ktorej detekcii sa body záujmu nachádzajú a podľa toho priraďujú detekciu k vzoru. V programe sú použité triedy `FeatureDetector` pre detekciu,`DescriptorExtractor` pre vytvorenie deskriptorov a `DescriptorMatcher` pre priradenie bodov záujmu. Objekty týchto typov sú používané pre detekciu metódami

```
Fdetector.detect(greyImage, imageKeyPoints)
```

, vytvorenie deskriptorov

```
Fdescriptor.compute(greyImage, imageKeyPoints,
   imageDescriptors)
```

a porovnávanie

```
matcher.match(detectionDescriptors,imageDescriptors ,
   matches)
```

Tieto metódy sú implementované z knižnice OpenCv pre jednotlivé objekty. V algoritme je taktiež použitý čiernobiely obraz, nakoľko je potrebné hľadať body záujmu ktoré sa vyskytujú hlavne v rohoch, ktoré model grayscale zachováva. Sledovanie prebieha podľa algoritmu 2.2 uvedeného vyššie.

2.3 Zlúčenie jednotlivých metód

2.3.1 Zlučovanie sekvenčným spôsobom, adaptívnosť sledovania

Po vytvorení rôznych metód sledovania objektu je potreba zlúčiť uvedené metódy tak, aby sledovanie riešilo problémy uvedené v 1.4.5. V programe je tento problém riešený uvedením sledovania do rôznych stavov, kedy je dominantný iný typ sledovania. Typ použitého sledovania závisí na udalostiach, ktoré sa odohrávajú v obraze. V prípade nastávajúcej oklúzie je dominantným detektorom `colorMatcher` na nastavený počet snímkov. Dominancia je vytvorená na základe nastavenia multiplikátora pre matcher ktorý sme zvolili. Vytvorený kód je vidieť nižšie.

```
if (oclusionHandler==0) {
  for (Template det : detectionList) {
    int counter=0;
    for (Template temp : detectionList) {
      if (temp.getDistance(det)<40) {
        counter++;
      }
    }
    if ((counter>=2)&&(oclusionHandler==0)) {
      oclusionHandler=15;
    }
  }
}
        if (oclusionHandler>0) {
  cMatcher.setMultiplicator(1.2);
  oclusionHandler--;
}
```

V tejto časti programu je možné nastaviť rôzne parametre, a to vzdialenosť kedy je dominantné sledovanie pomocou farby, počet sekvencií kedy je dominantné sledovanie pomocou farby a Bhattacharyya koeficient.

2.3.2 Paralelné zlučovanie sledovacích systémov

Je zlučovanie metód sledovania do uceleného výsledku. V programe je tento problém riešený pomocou matice priradenia ktorá je vypĺňaná jednotlivými sledovacými metódami. Jednotlivé metódy vypĺňajú túto maticu s rôznymi váhami, ktoré sú určené

pre potrebný typ sledovania. Pre správne sledovanie je potrebné tieto váhy dobre nastaviť. Pri zlučovaní taktiež dochádza k detekcií nového objektu, pokiaľ sa detekcia výrazne líši od objektov. Vytvorený kód môžeme vidieť nižšie

```
for (int i = 0; i < assignmentArray.length; i++)
   {
  double[] ds = assignmentArray[i];
  int idGoodMatch=200;
  double bestMatch=0;
  if (detectionList.get(i).
     getBhattacharyyaCoeficient()>2.0//nastavenie
      hodnot pri ktorých sa jedná a nový objekt
      || detectionList.get(i).
         getNearestSizeDistance()>80
      ///|| detectionList.get(i).
         getNearestSizeChange()>3000
      /*|| detectionList.get(i).
         getNumberOfGoodContours()==0*/)
  {
    listOfTemplates.add(detectionList.get(i));//
       Model zjavu
  }
  for (int j = 0; j < ds.length; j++) {//Priraď
     ovanie najlepšieho kandidáta
    if (bestMatch<ds[j]) {
      bestMatch=ds[j];
      idGoodMatch=j;
    }
  }
```

Taktiež sa v danom zlučovaní nachádza predikcia pohybu, ktorá sa zkladá z vytvorenia predikcie a korekcie a metódy sledovania pomocou predikovaného stavu. Túto metódu môžete vidieť nižšie.

```
public void match(List<Template> listOfTemplates, List<
   Template> listOfDetections, double[][] assignmentArray
   ) {
    // TODO Auto-generated method stub
    for (Template detection : listOfDetections) {
      int idGoodMatch=200;
      double nearestDistance=1000;
```

```
    Point detectionPoint = new Point();
    detectionPoint.x= (detection.getRect().x+detection.
       getRect().width)/2;
    detectionPoint.y=(detection.getRect().y+detection.
       getRect().height)/2;
    for (Template template : listOfTemplates) {
      double distance=Math.sqrt(Math.pow(template.
         getPredictedState().x-detectionPoint.x, 2)+
         Math.pow(template.getPredictedState().y-
         detectionPoint.y, 2));
    if (distance<nearestDistance) {
      nearestDistance=distance;
      idGoodMatch=listOfTemplates.indexOf(template);
    }
    }
    assignmentArray[listOfDetections.indexOf(detection)
       ][idGoodMatch]+=1*multiplicator;
    detection.setNearestSizeDistance(nearestDistance);
  }
}
```

3 VÝSLEDKY ŠTUDENTSKEJ PRÁCE

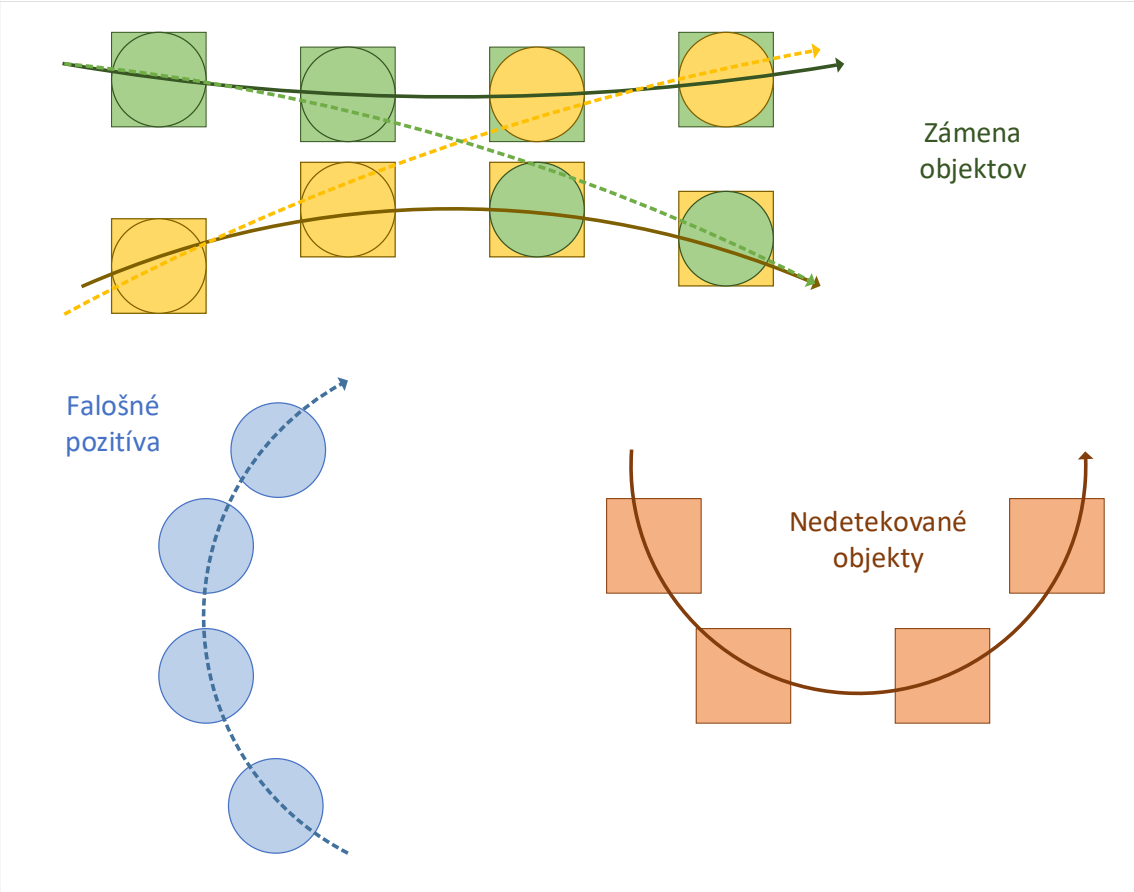
Táto kapitola sa zaoberá vhodným ohodnotením vytvoreného sledovacieho systému, ktorý zlučuje viaceré vlastnosti a porovnanie tohto systému so sledovacími systémami používajúcimi len jednu vlastnosť objektu. Obecne je sledovanie viacerých objektov ohodnotené pomocou metrík:

- MOTA - presnosť sledovania viacerých objektov
- MOTP - precíznosť sledovania viacerých objektov
- FP - celkový počet falošných pozitív (falošné detekcie)
- FN - celkový počet falošných negatív (nedetekované objekty)
- ID Sw. - celkový počet zámen objektov
- FPS - počet snímkov za sekundu

Nakoľko sa táto práca nezaoberá detekciou ale správnym priradením, je tento systém testovaný len z hľadiska presnosti sledovania, počtu snímkov za sekundu, celkového počtu zámen objektov a počtu zámien objektov za jeden snímok. U väčšiny menovaných metrík je výpočet zrozumiteľný z názvu avšak použitú metriku presnosti sledovania MOTA(z angl. multiple object tracking accuracy) je potrebné zadefinovať. Meranie precíznosti sledovania počítame ako:

$$MOTA = 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t}, \tag{3.1}$$

kde m_t je počet minutých objektov, mme_t je počet zámien objektov a fp_t je počet falošných pozitív, čiže počet detekcii ktoré nie sú požadovaným objektom[51]. g_t je počet objektov v snímku t. Nedetekované objekty, falošné pozitíva a zámeny objektov je možné vidieť na obr.3.1.



Obr. 3.1: Zlé priradenie objektov používané v precíznosti merania. Štvorec označuje objekt a kruh detekciu v štyroch sekvenciách.

Testovanie a nastavenie váh v praktickej časti prebiehalo na dvoch video sekvenciách, pričom prvá je získaná z [52] a druhá bola vytvorená pre rozšírenie sledovania tvárií. V oboch videosekvenciách sa nachádzajú tri osoby, ktoré prechádzajú cez seba plynulou rýchlosťou, vychádzajú a vchádzajú do obrazu alebo sa otáčajú v priestore, čím zaniká detekcia tváre. Výsledky sledovania s paralelným zlučovacím systémom boli porovnané so sledovacími systémami, ktoré používali priradenie pomocou jedného typu sledovania a boli relatívne spoľahlivé. Za spoľahlivé metódy sledovania môžme považovať sledovanie pomocou farby a vzdialenosti, nakoľko je u nich nízky počet zámien objektov. Výsledky sledovania tvárií je možno vidieť v tabuľke 3.1 a 3.2. Pre sledovanie tvárí bol použitý kaskádový Haar detektor.

Tab. 3.1: Úspešnosť sledovania pomocou rôznych metód pre prvú videosekvenciu

Typ sledovania	MOTA-precíznosť sledovania[%]	ID sw.-zmena identity[-]	FPS
Histogramy	94,2177	9	5.175719
Vzdialenosť	94,3027	8	5.311475
Zlúčenie metód	94,5578	0	5.005149
Bez adaptívnosti	94,5578	0	5,169851

Tab. 3.2: Úspešnosť sledovania pomocou rôznych metód pre druhú videosekvenciu

Typ sledovania	MOTA-precíznosť sledovania[%]	ID sw.-zmena identity[-]	FPS
Histogramy	92,0548	2	7.548747
Vzdialenosť	91,3699	7	7.384196
Zlúčenie metód	92,3288	0	7.038961
Bez adaptívnosti	92,0548	2	7,245989

Okrem sledovania tvárií bolo ohodnotené sledovanie hokejistov v pohyblivom videu [53] a sledovanie chodcov v [54], pri ponechaní nastavenia systému z predošlých videí a použití detektoru histogramu orientovaných gradientov. Výsledky je možno vidieť v tabuľke 3.3 a 3.4.

Tab. 3.3: Úspešnosť sledovania pomocou rôznych metód pre tretiu videosekvenciu

Typ sledovania	MOTA-precíznosť sledovania[%]	ID sw.-zmena identity[-]	FPS
Histogramy	51,0606	16	10,96639
Vzdialenosť	54,0909	36	10,65306
Zlúčenie metód	54,3000	15	10,44418
Bez adaptívnosti	0,539394	17	10,74074

Tab. 3.4: Úspešnosť sledovania pomocou rôznych metód pre štvrtú videosekvenciu

Typ sledovania	MOTA-precíznosť sledovania[%]	ID sw.-zmena identity[-]	FPS
Histogramy	73,8574	47	8,386235
Vzdialenosť	73,309	56	8,249153
Zlúčenie metód	73,4918	50	8,063688
Bez adaptívnosti	0,731871	58	8,253968

Z dosiahnutých výsledkov je možné vyvodiť, že zlúčenie rôznych sledovacích metód zlepšuje úspešnosť detekcie. Taktiež zlúčené systémy nemajú veľkú výpočtovú

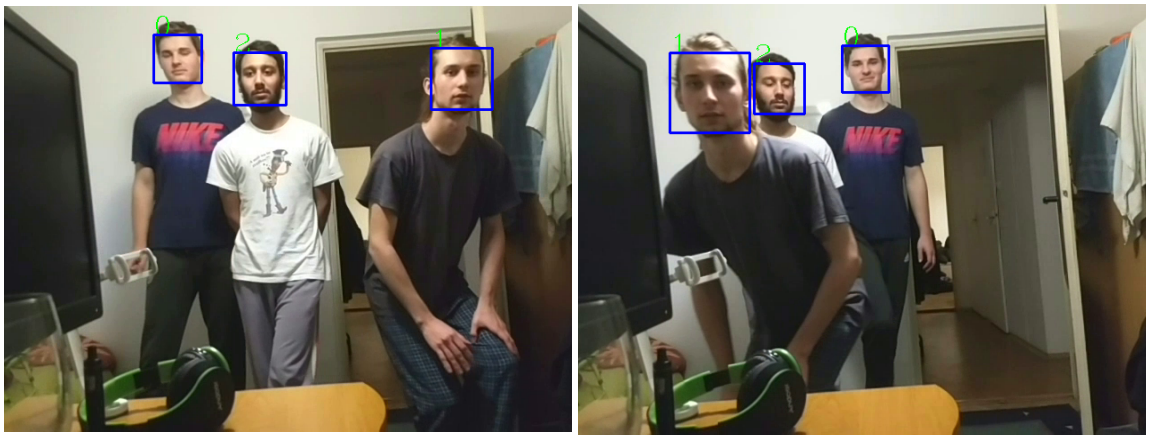
zložitosť avšak tento systém nie je možné použiť pre spracovanie obrazu v reálnom čase, nakoľko nepresahuje 15FPS. Zo sledovania osôb sa dá preukázať aké je dôležité nastaviť správne váhy pre jednotlivé metódy a adaptívnosť sledovania. Ďalším spôsobom, ako je možné zhodnotiť výsledky na základe riešení problémov spomenutých v 1.4.5, pričom hlavným cieľom tejto práce je riešenie oklúzie dvoch objektov. Tento problém bol riešený vytvorením adaptívnosti systému ktorý podľa tabuliek napomohol k lepšiemu sledovaniu. Riešenie oklúzie je znázornené na obr.3.2,3.3,3.5 a 3.4.

a) b)

Obr. 3.2: a)Obraz v sekvencii pred oklúziou sledovaných objektov b)Obraz v sekvencii po oklúzii so sledovaným objektom

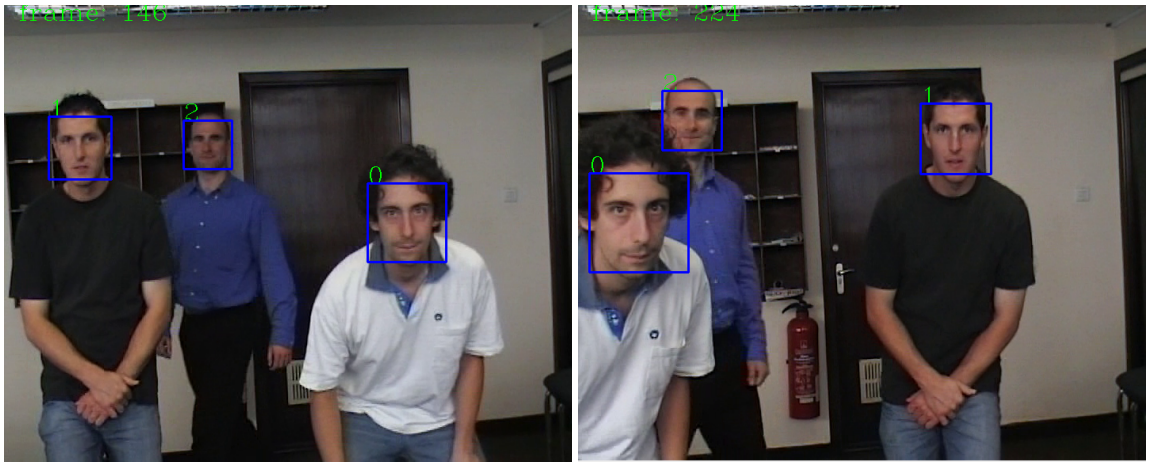
a) b)

Obr. 3.3: a)Obraz v sekvencii pred oklúziou sledovaných objektov b)Obraz v sekvencii po oklúzii so sledovaným objektom



a) b)

Obr. 3.4: a)Obraz v sekvencii pred rýchlou oklúziou sledovaných objektov b)Obraz v sekvencii po rýchlej oklúzii so sledovaným objektom



a) b)

Obr. 3.5: a)Obraz v sekvencii pred oklúziou a východom z obrazu sledovaných objektov b)Obraz v sekvencii po oklúzii so sledovaným objektom

Taktiež môžeme porovnať sledovací systém pomocou videí z MOTA [55] a [56]. Pre porovnanie boli použité sledovacie systémy [57], [58], [59], [60], [61] a [62].

Tab. 3.5: Porovnanie sledovacích systémov na videu TUD-Crossing z MOTA benchmarku

Sledovací systém	MOTA[%]	ID sw.[-]	FP	FN	Hardware
AP_HWDPL_p	61.3	12	14	401	GTX 1080, 2.3 GHz, 1 Core
AMIR15	73.7	16	18	256	TITAN X 3GHZ 1 Core
HybridDAT	73.3	8	42	244	3.4 GHZ, 1 Core
TDAM	55.4	8	52	431	3.4 GHZ, 1 Core
LFNF	64.6	28	14	348	3.3 GHZ, 2 Core
TBX	58.3	28	150	282	3.50GHz, 1 Core
Naša metóda	67,15	26	118	129	2.0GHz, 4 Core

Tab. 3.6: Porovnanie sledovacích systémov na videu ETH-Crossing z MOTA benchmarku

Sledovací systém	MOTA[%]	ID sw.[-]	FP	FN	Hardware
AP_HWDPL_p	34.2	10	20	630	GTX 1080, 2.3 GHz, 1 Core
AMIR15	32.9	21	33	619	TITAN X 3GHZ 1 Core
HybridDAT	24.1	1	29	731	3.4 GHZ, 1 Core
TDAM	25.1	2	18	731	3.4 GHZ, 1 Core
LFNF	29.6	3	3	700	3.3 GHZ, 2 Core
TBX	20.1	6	35	760	3.50GHz, 1 Core
Naša metóda	24.12	34	261	584	2.0GHz, 4 Core

Z týchto výsledkov je možno ukázať že nami vytvorený sledovací systém je porovnateľný s vybranými systémami z MOTA benchmarku. Taktiež má sledovací systém lepšiu detekciu objektov, avšak na úkor falošných pozitív. Falošné pozitíva taktiež poškodzujú priradenie detekcí k objektom a tým dochádza k zámene identít.

4 ZÁVER

Táto práca sa zaoberá tvorbou neučiacich sa algoritmov pre sledovanie objektov vo videu. Cieľom bolo vytvoriť algoritmus, ktorý pri sledovaní objektu používa viaceré jeho parametre. Ďalším krokom práce bolo riešenie priraďovacieho problému a zoznámenie sa s problémami vznikajúcimi pri sledovaní objektov.

V prvej kapitole sú popísané metódy, ktoré sú neskôr použité v praktickej časti a metódy, ktoré sa v súčasnej dobe používajú na sledovanie objektov. Použitie rôznych metód sledovania vedie k zníženiu chybovosti, čím prispieva k zlepšeniu sledovania. V druhej kapitole sa rieši problém priraďovania, ktorá je venovaná hlavne maďarskej metóde priraďovania. Problém priraďovania rieši vyhodnotenie priradenia rôznych sledovacích metód, ktoré nedospeli k rovnakému výsledku. V tretej kapitole sú uvedené hlavné problémy vyskytujúce sa pri sledovaní sledovaní objektov v obraze. Táto časť sa venuje vysvetleniu problematiky a jeho riešením. Nasledujúca kapitola sa venuje praktickému riešeniu programu. Tu je opísaný návrh hlavných tried používaných v rámci sledovania, algoritmus sledovacích metód a zlúčenie týchto metód pre priradenie objektu.

Hlavným prínosom práce je použitie rôznych metód pre sledovanie objektov, ktoré vedie ku spresneniu sledovania objektov. V rámci tejto časti je vyhodnotená presnosť sledovania pomocou benchmarku MOTA a vyhodnotenie riešenia problému oklúzie objektov. Taktiež je v tejto časti porovnanie úspešnosti rozličných metód sledovania zvlášť a spolu. Testovanie sledovania bolo vykonávané na základe viacerých videosekvencí a vyhodnotené výsledky boli spracované v tabuľkách.

LITERATÚRA

[1] Karl Granström and Marcus Baum. Extended object tracking: Introduction, overview and applications. *CoRR*, abs/1604.00970, 2016. URL: `http://arxiv.org/abs/1604.00970`, `arXiv:1604.00970`.

[2] Wilhelm Burger and Mark J Burge. *Digital image processing: an algorithmic introduction using Java*. Springer, 2016.

[3] Fouzi Douak, Redha Benzid, and Nabil Benoudjit. Color image compression algorithm based on the dct transform combined to an adaptive block scanning. *AEU-International Journal of Electronics and Communications*, 65(1):16–26, 2011.

[4] John Canny. A computational approach to edge detection. In *Readings in Computer Vision*, pages 184–203. Elsevier, 1987.

[5] Frank J Aherne, Neil A Thacker, and Peter I Rockett. The bhattacharyya metric as an absolute similarity measure for frequency coded data. *Kybernetika*, 34(4):363–368, 1998.

[6] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Real-time tracking of non-rigid objects using mean shift. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 142–149. IEEE, 2000.

[7] Dorin Comaniciu and Peter Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 24(5):603–619, 2002.

[8] Carlo Tomasi and Takeo Kanade. Detection and tracking of point features. 1991.

[9] Jean-Yves Bouguet. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel Corporation*, 5(1-10):4, 2001.

[10] Xin Li, Kejun Wang, Wei Wang, and Yang Li. A multiple object tracking method using kalman filter. In *Information and Automation (ICIA), 2010 IEEE International Conference on*, pages 1862–1866. IEEE, 2010.

[11] SA Vigus, David R Bull, and Cedric Nishan Canagarajah. Video object tracking using region split and merge and a kalman filter tracking algorithm. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, volume 1, pages 650–653. IEEE, 2001.

[12] Rudolph Emil Kalman et al. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45, 1960.

[13] Alper Yilmaz, Xin Li, and Mubarak Shah. Object contour tracking using level sets. In *Asian Conference on Computer Vision*, 2004.

[14] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE, 2011.

[15] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001.

[16] Andrew Witkin. Scale-space filtering: A new approach to multi-scale description. In *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'84.*, volume 9, pages 150–153. IEEE, 1984.

[17] Shoujia Wang, Wenhui Li, Ying Wang, Yuanyuan Jiang, Shan Jiang, and Ruilin Zhao. An improved difference of gaussian filter in face recognition. *Journal of Multimedia*, 7(6):429–433, 2012.

[18] Exploiting eigenvalues of the hessian matrix for volume decimation. 2001.

[19] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[20] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[21] Changjiang Yang, Ramani Duraiswami, and Larry Davis. Fast multiple object tracking via a hierarchical particle filter. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 212–219. IEEE, 2005.

[22] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 24(7):971–987, 2002.

[23] Wenchao Zhang, Shiguang Shan, Wen Gao, Xilin Chen, and Hongming Zhang. Local gabor binary pattern histogram sequence (lgbphs): a novel non-statistical model for face representation and recognition. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 786–791. IEEE, 2005.

[24] Chao He, Yuan F Zheng, and Stanley C Ahalt. Object tracking using the gabor wavelet transform and the golden section algorithm. *IEEE transactions on multimedia*, 4(4):528–538, 2002.

[25] Lance M Kaplan, Yaakov Bar-Shalom, and William D Blair. Assignment costs for multiple sensor track-to-track association. *IEEE Transactions on Aerospace and Electronic Systems*, 44(2), 2008.

[26] Ingemar J Cox, Matt L Miller, Roy Danchick, and GE Newnam. A comparison of two algorithms for determining ranked assignments with application to multitarget tracking and motion correspondence. *IEEE Transactions on Aerospace and Electronic Systems*, 33(1):295–301, 1997.

[27] Harold W Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics (NRL)*, 2(1-2):83–97, 1955.

[28] Eliane Maria Loiola, Nair Maria Maia de Abreu, Paulo Oswaldo Boaventura-Netto, Peter Hahn, and Tania Querido. A survey for the quadratic assignment problem. *European journal of operational research*, 176(2):657–690, 2007.

[29] Bjorn Stenger, Thomas Woodley, and Roberto Cipolla. Learning to track with multiple observers. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2647–2654. IEEE, 2009.

[30] Lin Ma, Jiwen Lu, Jianjiang Feng, and Jie Zhou. Multiple feature fusion via weighted entropy for visual tracking. In *Proceedings of the IEEE international conference on computer vision*, pages 3128–3136, 2015.

[31] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 1, pages I–I. IEEE, 2002.

[32] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

[33] Eric W Weisstein. Normalized vector. *From MathWorld–A Wolfram Web Resource. http://mathworld. wolfram. com/NormalizedVector. html*.

[34] Jie Cao, Leilei Guo, Jinhua Wang, and Di Wu. Object tracking based on multiple features adaptive fusion. *Indonesian Journal of Electrical Engineering and Computer Science*, 12(7):5621–5628, 2014.

[35] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[36] CV Open. Opencv documentation, 2013.

[37] Jeffrey Ho, Kuang-Chih Lee, Ming-Hsuan Yang, and David Kriegman. Visual tracking using learned linear subspaces. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2004.

[38] Douglas Gray, Shane Brennan, and Hai Tao. Evaluating appearance models for recognition, reacquisition, and tracking. In *Proc. IEEE International Workshop on Performance Evaluation for Tracking and Surveillance (PETS)*, number 5, pages 1–7. Citeseer, 2007.

[39] Shaohua Kevin Zhou, Rama Chellappa, and Baback Moghaddam. Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Transactions on Image Processing*, 13(11):1491–1506, 2004.

[40] Allan D Jepson, David J Fleet, and Thomas F El-Maraghi. Robust online appearance models for visual tracking. *IEEE transactions on pattern analysis and machine intelligence*, 25(10):1296–1311, 2003.

[41] Darrel Greenhill, J Renno, James Orwell, and Graeme A Jones. Occlusion analysis: Learning and utilising depth maps in object tracking. *Image and Vision Computing*, 26(3):430–441, 2008.

[42] Hanxuan Yang, Ling Shao, Feng Zheng, Liang Wang, and Zhan Song. Recent advances and trends in visual tracking: A review. *Neurocomputing*, 74(18):3823–3831, 2011.

[43] Sachiko Iwase and Hideo Saito. Parallel tracking of all soccer players by integrating detected positions in multiple view images. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 4, pages 751–754. IEEE, 2004.

[44] Dorin Comaniciu and Visvanathan Ramesh. Mean shift and optimal prediction for efficient object tracking. In *Image processing, 2000. proceedings. 2000 international conference on*, volume 3, pages 70–73. IEEE, 2000.

[45] Bohyung Han, Seong-Wook Joo, and Larry S Davis. Probabilistic fusion tracking using mixture kernel-based bayesian filtering. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[46] Wei Du and Justus Piater. A probabilistic approach to integrating multiple cues in visual tracking. In *European Conference on Computer Vision*, pages 225–238. Springer, 2008.

[47] The Eclipse Foundation. http://www.eclipse.org/l, 2017. URL: http://www.eclipse.org/.

[48] OpenCV team. About. https://opencv.org/about.html, 2017. URL: https://opencv.org/about.html.

[49] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking:a survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[50] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[51] Keni Bernardin and Rainer Stiefelhagen. Evaluating multiple object tracking performance: the clear mot metrics. *Journal on Image and Video Processing*, 2008:1, 2008.

[52] Emilio Maggio, Elisa Piccardo, Carlo Regazzoni, and Andrea Cavallaro. Particle phd filtering for multi-target visual tracking. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 1, pages I–1101. IEEE, 2007.

[53] Jessica Politi. Men playing basketball, 2016. URL: https://videos.pexels.com/videos/men-playing-basketball-992625.

[54] Shakr.com. Ice hockey, 2016. URL: https://videos.pexels.com/videos/ice-hockey-857637.

[55] Andreas Ess, Bastian Leibe, and Luc Van Gool. Depth and appearance for mobile scene analysis. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[56] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. People-tracking-by-detection and people-detection-by-tracking. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[57] Long Chen, Haizhou Ai, Chong Shang, Zijie Zhuang, and Bo Bai. Online multi-object tracking with convolutional neural networks. In *Image Processing (ICIP), 2017 IEEE International Conference on*, pages 645–649. IEEE, 2017.

[58] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. Tracking the untrackable: Learning to track multiple cues with long-term dependencies. *arXiv preprint arXiv:1701.01909*, 4(5):6, 2017.

[59] Min Yang, Yuwei Wu, and Yunde Jia. A hybrid data association framework for robust online multi-object tracking. *IEEE Transactions on Image Processing*, 26(12):5667–5679, 2017.

[60] Min Yang and Yunde Jia. Temporal dynamic appearance modeling for online multi-person tracking. *Computer Vision and Image Understanding*, 153:16–28, 2016.

[61] Hao Sheng, Li Hao, Jiahui Chen, Yang Zhang, and Wei Ke. Robust local effective matching model for multi-target tracking. In *Pacific Rim Conference on Multimedia*, pages 233–243. Springer, 2017.

[62] Roberto Henschel, Laura Leal-Taixé, Bodo Rosenhahn, and Konrad Schindler. Tracking with multi-level features. *arXiv preprint arXiv:1607.07304*, 2016.

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

SIFT	Škálovo invariantná transformácia vlastnosti- detektor a deskriptor bodov záujmu
ORB	Oriented FAST rotated BRIEF-detektor a deskriptor bodov záujmu
KLT	Kanade-Lucas-Tomasi-sledovacia metóda pomocou extrakcie vlastností
SURF	Speeded up robust features- detektor a deskriptor lokálnej extrakcie vlastností
DOG	Difference of Gaussians-vylepšenie SIFT detektoru pomocou diferencie oblastí bodov záujmu filtrovaných Gausovým filtrom
MOTA	presnosť sledovania viacerých objektov(z angl. multiple object tracking accuracy)
MOTA	precíznosť sledovania viacerých objektov(z angl. multiple object tracking precision)
FP	celkový počet falošných pozitív (falošné detekcie)
FN	celkový počet falošných negatív (nedetekované objekty)
ID Sw.	celkový počet zámen objektov vo videu
FPS	počet snímkov za sekundu

ZOZNAM PRÍLOH

A OBSAH PRILOŽENÉHO CD

Na priloženom médiu sa nachádza digitálna verzia bakalárskej práce a program prakticej časti. Programová časť bola realizovaná vo vývojovom prostredí Eclipse Jee Oxygen spolu s knižnicou OpenCv 2.4.13, ktorá sa nachádza v adresári programu. Súbory ožnačené * vznikajú až po spustení programu. Taktiež je potrebné pridať knižnicu OpenCv 2.4.13.6 pre javu do zložky `lib` zo stránky `https://opencv.org/releases.html`. Nastavenie detekovaného videa a výber klasifikátorov je možné v hlavnej triede HumanDetector.class na riadku 59 a 73. Taktiež je možné nestaviť váhy jednotlivých systémov v ich triede pomocou premennej `multiplicator`.

- `/186035` ..... koreňový adresár priloženého CD
  - `sablona-prace.pdf` ..... pdf verzia bakalárskej práce
  - `BP_18_ObjectTracking_Boszorad` ..... priečinok programovej časti
    - `.settings`
      - `org.eclipse.jdt.core.prefs`
      - `org.eclipse.ltk.core.refactoring.prefs`
    - `bin`
      - `cz/vutbr/feec/utko/objectTracking`
        - `example`
          - `ColorMatcher.class` ..... trieda priradenia podľa histogramu
          - `ContourMatcher.class` ..... trieda priradenia podľa kontúr
          - `DistanceMatcher.class` ..... trieda priradenia podľa vzdialenosti
          - `HumanDetector.class` ..... trieda pre spustenie
          - `IMatcher.class` ..... rozhranie pre triedy priradenia
          - `PredictionMatcher.class` ..... trieda priradenia podľa predikcie
          - `SizeMatcher.class` ..... trieda priradenia podľa veľkosti objektu
          - `Template.class` ..... trieda reprezentujúca detekcie
        - `helpers`
          - `ImageFunctions.class`
    - `data`
      - `test_video.avi` ..... testovacie video
      - `opencv_detectors`
        - `haarcascades`
          - `haarcascade_frontalface_alt.xml` . haarove kaskády pre detekciu tváre
        - `hogcascades`
          - `hogcascade_pedestrians.xml` ... hog kaskády pre detekciu chodcov
    - `lib`
      - `java` ..... potrebné knižnice
        - `opencv2413.jar` ..... knižnica opencv
        - `x64`
          - `opencv_java2413.dll`
        - `x86`
          - `opencv_java2413.dll`
    - `src`

- cz/vutbr/feec/utko/objectTracking
  - example
    - ColorMatcher.java
    - ContourMatcher.java
    - DistanceMatcher.java
    - HumanDetector.java
    - IMatcher.java
    - PredictionMatcher.java
    - SizeMatcher.java
    - Template.java
  - helpers
    - ImageFunctions.java
- .classpath.classpath
- .project.project
- opencv_ffmpeg2413_64.dll ..... súbor pre používanie kodekov
- *test.txt ..... textový záznam výsledkov sledovania